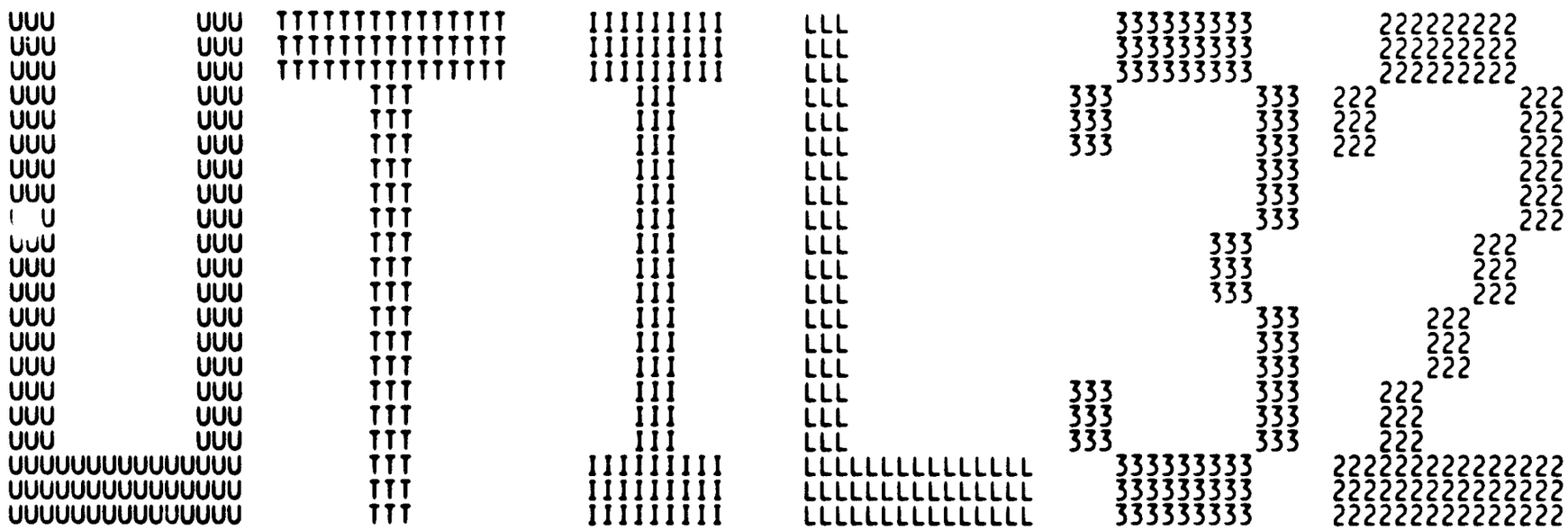

UUU UUU TTTTTTTTTTTTTTT IIIIIIIII LLL 333333333 222222222
UUU UUU TTTTTTTTTTTTTTT IIIIIIIII LLL 333333333 222222222
UUU UUU TTTTTTTTTTTTTTT IIIIIIIII LLL 333333333 222222222
UUU UUU TTT III LLL 333 333 222 222
UUU UUU TTT III LLL 333 333 222 222
UUU UUU TTT III LLL 333 333 222 222
UUU UUU TTT III LLL 333 222
UUU UUU TTT III LLL 333 222
UUU UUU TTT III LLL 333 222
UUU UUU TTT III LLL 333 222
UUU UUU TTT III LLL 333 222
UUU UUU TTT III LLL 333 222
UUU UUU TTT III LLL 333 222
UUU UUU TTT III LLL 333 222
UUU UUU TTT III LLL 333 222
UUU UUU TTT III LLL 333 222
UUU UUU TTT III LLL 333 333 222
UUUUUUUUUUUUUUUU TTT IIIIIIIII LLLLLLLLLLLLLLL 333333333 222222222222222
UUUUUUUUUUUUUUUU TTT IIIIIIIII LLLLLLLLLLLLLLL 333333333 222222222222222
UUUUUUUUUUUUUUUU TTT IIIIIIIII LLLLLLLLLLLLLLL 333333333 222222222222222

```
  SSSSSSSS  EEEEEEEEEE    AAAAAA    RRRRRRRR      CCCCCCCC  HH      HH
  SSSSSSSS  EEEEEEEEEE    AAAAAA    RRRRRRRR      CCCCCCCC  HH      HH
SS          EE          AA      AA  RR      RR  CC          HH      HH
SS          EE          AA      AA  RR      RR  CC          HH      HH
SS          EE          AA      AA  RR      RR  CC          HH      HH
SS          EE          AA      AA  RR      RR  CC          HH      HH
  SSSSSS    EEEEEEEE    AA      AA  RRRRRRRR    CC          HHHHHHHHHH
  SSSSSS    EEEEEEEE    AA      AA  RRRRRRRR    CC          HHHHHHHHHH
        SS  EE          AAAAAAAAAA  RR  RR      CC          HH      HH
        SS  EE          AAAAAAAAAA  RR  RR      CC          HH      HH
        SS  EE          AA      AA  RR    RR    CC          HH      HH
        SS  EE          AA      AA  RR    RR    CC          HH      HH
SSSSSSSS    EEEEEEEEEE  AA      AA  RR      RR    CCCCCCCC  HH      HH
SSSSSSSS    EEEEEEEEEE  AA      AA  RR      RR    CCCCCCCC  HH      HH


LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
;  1     0001  0 MODULE search (                ! Search file(s) for a string utility
;  2     0002  0                                IDENT = 'V04-000',
;  3     0003  0                                MAIN = main
;  4     0004  0                                ) =
;  5     0005  1 BEGIN
;  6     0006  1
;  7     0007  1 !
;  8     0008  1 !*****************************************************************************
;  9     0009  1 !*                                                                           *
; 10     0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
; 11     0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
; 12     0012  1 !*  ALL RIGHTS RESERVED.                                                     *
; 13     0013  1 !*                                                                           *
; 14     0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
; 15     0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
; 16     0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
; 17     0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
; 18     0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
; 19     0019  1 !*  TRANSFERRED.                                                             *
; 20     0020  1 !*                                                                           *
; 21     0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
; 22     0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
; 23     0023  1 !*  CORPORATION.                                                             *
; 24     0024  1 !*                                                                           *
; 25     0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
; 26     0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
; 27     0027  1 !*                                                                           *
; 28     0028  1 !*                                                                           *
; 29     0029  1 !*****************************************************************************
; 30     0030  1
; 31     0031  1 !++
; 32     0032  1 ! FACILITY:  SEARCH
; 33     0033  1 !
; 34     0034  1 ! ABSTRACT:
; 35     0035  1 !
; 36     0036  1 !       This utility program searchs a file or files for a specified
; 37     0037  1 !       string and lists all occurrences.
; 38     0038  1 !
; 39     0039  1 ! ENVIRONMENT:
; 40     0040  1 !
; 41     0041  1 !       VAX/VMS operating system, unprivileged user mode utility,
; 42     0042  1 !       operates at non-AST level.
; 43     0043  1 !
; 44     0044  1 ! AUTHOR:  Len Kawell,  CREATION DATE:  22 Dec 1978
; 45     0045  1 !
; 46     0046  1 ! Modified by:
; 47     0047  1 !
; 48     0048  1 !       V03-004 CWH3004         CW Hobbs                9-Aug-1984
; 49     0049  1 !               Change output formatting routines to support eight-
; 50     0050  1 !               bit characters.
; 51     0051  1 !
; 52     0052  1 !       V03-003 AEW0001         Anne E. Warner          1-May-1984
; 53     0053  1 !               Add support for search lists.
; 54     0054  1 !                 - remove related name block from RMS definitions.
; 55     0055  1 !                 - add argument SCAN_CONTEXT to LIBSFILE_SCAN for
; 56     0056  1 !                   sticky search lists.
; 57     0057  1 !
```

```
;   58          0058  1 !       V03-002 CWH0015         CW Hobbs                8-Sep-1982
;   59          0059  1 !               Enable /STATISTICS
;   60          0060  1 !
;   61          0061  1 !       V03-001 CWH0014         CW Hobbs                22-Mar-1982
;   62          0062  1 !               Change WRITEERR to SEVERE so that only one message is
;   63          0063  1 !               printed when unable to write the output.
;   64          0064  1 !
;   65          0065  1 !       V02-013 CWH0013         CW Hobbs                20-Feb-1982
;   66          0066  1 !               Add nambufs for output file so that the file name is
;   67          0067  1 !               available for error messages.  Change output segmenting
;   68          0068  1 !               to avoid record too big errors.
;   69          0069  1 !
;   70          0070  1 !       V02-012 CWH0012         CW Hobbs                2-Feb-1982
;   71          0071  1 !               Use LIB$FILE_SCAN instead of $PARSE and $SEARCH to
;   72          0072  1 !               perform like other DCL commands.
;   73          0073  1 !
;   74          0074  1 !       V02-011 CWH0009         CW Hobbs                25-Jan-1982
;   75          0075  1 !               Make error messages for insufficient virtual memory
;   76          0076  1 !               more meaningful.
;   77          0077  1 !
;   78          0078  1 !       V02-010 CWH0008         CW Hobbs                18-Jan-1982
;   79          0079  1 !               Use shared messages where possible, add WRITEERR message.
;   80          0080  1 !
;   81          0081  1 !       V02-009 CWH0007         CW Hobbs                14-Dec-1981
;   82          0082  1 !               Fix loop if no directory in resultant string for EXCLUDE.
;   83          0083  1 !
;   84          0084  1 !       V02-008 CWH0006         CW Hobbs                13-Dec-1981
;   85          0085  1 !               Remove dummy routine.  Do $TRNLOG on /EXCLUDE file
;   86          0086  1 !               names.  Correct problem with /WINDOW=0.  Remove
;   87          0087  1 !               some code for testing.
;   88          0088  1 !
;   89          0089  1 !       V02-007 CWH0004         CW Hobbs                30-Nov-1981
;   90          0090  1 !               Speedups, make /STATISTICS a conditional assembly
;   91          0091  1 !               option.  Restructure for expression matches.
;   92          0092  1 !
;   93          0093  1 !       V02-006 CWH0002         CW Hobbs                28-Oct-1981
;   94          0094  1 !                 Change FOR$CNV_IN_I to OTS$CVT_TI_L entry.  Add more
;   95          0095  1 !               error codes.  Remove MBC from input FAB, use process
;   96          0096  1 !               defaults.  Remove filename header from between windows and
;   97          0097  1 !               change window separator to '********'.  Put output record in
;   98          0098  1 !               several segments if longer than mrs of output file, also
;   99          0099  1 !               check status of rms_put operation.
;  100          0100  1 !                 Make search-string input parameter a list, move match
;  101          0101  1 !               check to separate routine, add /MATCH qualifier to
;  102          0102  1 !               set sense of match.
;  103          0103  1 !                 Redo windowing.  Save vector of RFA's rather than records.
;  104          0104  1 !               Use locate mode exclusively, set SQO only if no prev recs.
;  105          0105  1 !               Replace /ADDITIONAL and /PREVIOUS with /WINDOW qualifier.
;  106          0106  1 !               Print filename only if /WINDOW=0 is specified.
;  107          0107  1 !                 Improve error messages, especially I/O error signalling.
;  108          0108  1 !               Change truncate error to one per file, not one per record.
;  109          0109  1 !                 Other new qualifiers:
;  110          0110  1 !                       /EXACT      - use exact case matches
;  111          0111  1 !                       /EXCLUDE    - omit list of files from search
;  112          0112  1 !                       /FORMAT     - convert control chars unless disabled
;  113          0113  1 !                       /HEADING    - give control of printing of heading
;  114          0114  1 !                       /LOG        - signal filenames searched
```

```
;  115          0115  1 !                     /NUMBERS    - print source line #'s
;  116          0116  1 !                     /REMAINING  - print from match to EOF
;  117          0117  1 !                     /STATISTICS - print some access information
;  118          0118  1 !
;  119          0119  1 !       V02-005 CWH0001         CW Hobbs                24-Aug-1981
;  120          0120  1 !               Define external routines as general addressing mode.
;  121          0121  1 !               Display the filespec header whenever a list of
;  122          0122  1 !               input filenames is given.
;  123          0123  1 !
;  124          0124  1 !       V004    TMH0004         Tim Halvorsen           29-Jun-1981
;  125          0125  1 !               Increase size of input buffer.  Open input file for
;  126          0126  1 !               shared read/write access.  Do not display the filespec
;  127          0127  1 !               header if non-wild search.
;  128          0128  1 !
;  129          0129  1 !       V02-003 CNH0032         Chris Hume              16-Apr-1980
;  130          0130  1 !               Added windowing and (new) CLI callback.
;  131          0131  1 !
;  132          0132  1 !       V02-002                 Len Kawell              14-Mar-1980
;  133          0133  1 !               General cleanup, fixups, and added command line parameters.
;  134          0134  1 !
;  135          0135  1 !       V02-001                 B.L. Schreiber          18-May-1979
;  136          0136  1 !               Add LIB$SIGNAL to external routines.
;  137          0137  1 !
;  138          0138  1 !--
```

```
;  140          0139  1 !
;  141          0140  1 ! Include files
;  142          0141  1 !
;  143          0142  1 LIBRARY
;  144          0143  1     'SYS$LIBRARY:STARLET';              ! VAX/VMS common definitions
;  145          0144  1
;  146          0145  1 !
;  147          0146  1 ! Define the compile-time switches used for conditional assemblies
;  148          0147  1 !
;  149          0148  1 LITERAL
;  150          0149  1     switch_statistics   = 1;            ! Include the /STATISTICS code
;  151          0150  1 !
;  152          0151  1 ! Structure declarations
;  153          0152  1 !
;  154          0153  1 STRUCTURE
;  155          0154  1
;  156          0155  1         bblockvector [I, O, P, S, E; N, BS] =
;  157          0156  1                 [N*BS]
;  158          0157  1                 ((bblockvector+I*BS)+0)<P,S,E>;
;  159          0158  1 !
;  160          0159  1 ! Macros
;  161          0160  1 !
;  162          0161  1 MACRO
;  163          0162  1
;  164          0163  1     dsc_long1 = 0,0,32,0 %,             ! First longword of descriptor
;  165          0164  1     dsc_dynam = dsc$k_class_d^24 OR dsc$k_dtype_t^16 %,  ! dynamic desc
;  166          0165  1
;  167          0166  1     ! Define shorthand for a single initialized dynamic string desc
;  168          0167  1     !
;  169        M 0168  1     dyn_str_desc = BLOCK [dsc$k_d_bln,BYTE]
;  170        M 0169  1                         PRESET( [dsc$b_class] = dsc$k_class_d,
;  171          0170  1                                 [dsc$b_dtype] = dsc$k_dtype_t) %,
;  172          0171  1
;  173          0172  1     ! Define macro for a single initialized static string desc.
;  174          0173  1     !
;  175        M 0174  1     stat_str_desc (L, A) = BLOCK [dsc$k_s_bln,BYTE]
;  176        M 0175  1                         PRESET( [dsc$b_class]   = dsc$k_class_s,
;  177        M 0176  1                                 [dsc$b_dtype]   = dsc$k_dtype_t,
;  178        M 0177  1                                 [dsc$w_length]  = (L),
;  179          0178  1                                 [dsc$a_pointer] = (A) ) %,
;  180          0179  1
;  181          0180  1     ! Define different psects for initialized and demand-zero storage.
;  182          0181  1     !
;  183          0182  1     OWN_CRF     = PSECT OWN    = $OWN$_CRF     (WRITE); OWN    %,
;  184          0183  1     OWN_DZRO    = PSECT OWN    = $OWN$_DZRO    (WRITE); OWN    %;
;  185          0184  1 !
;  186          0185  1 ! Define special linkages for very frequently called routines
;  187          0186  1 !
;  188          0187  1 LINKAGE
;  189          0188  1     JSB_FORACTDUM = JSB (REGISTER=2,REGISTER=3,REGISTER=0)
;  190          0189  1                         : NOPRESERVE(2,3,4,5)   PRESERVE(6,7,8,9)  NOTUSED(10,11),
;  191          0190  1     JSB_FORACTTEX = JSB (REGISTER=4,REGISTER=5,REGISTER=2)
;  192          0191  1                         : NOPRESERVE(2,3,4,5)   PRESERVE(6,7,8,9)  NOTUSED(10,11),
;  193          0192  1     JSB_FORPUT    = JSB (REGISTER=2,REGISTER=3,REGISTER=4)
;  194          0193  1                         : NOPRESERVE(2,3,4)   PRESERVE(5,6,7,8,9)  NOTUSED(10,11),
;  195          0194  1     JSB_GETNEXREC = JSB (REGISTER=3,REGISTER=4)
;  196          0195  1                         : NOPRESERVE(2)         NOTUSED(5,6,7,8,9,10,11),
```

```
;  197     0196  1      JSB_PUTOUT    = JSB (REGISTER=2,REGISTER=3)
;  198     0197  1                          : NOPRESERVE(2,3)        NOTUSED(6,7,8,9,10,11),
;  199     0198  1      JSB_STDMAT    = JSB (REGISTER=4,REGISTER=5)
;  200     0199  1                          : NOPRESERVE(2,3)        NOTUSED(9,10,11),
;  201     0200  1      JSB_UPCASE    = JSB (REGISTER=1, REGISTER=2, REGISTER=3)
;  202     0201  1                          : NOPRESERVE(2,3)        NOTUSED(4,5,6,7,8,9,10,11);
;  203     0202  1 !
;  204     0203  1 ! Table of contents
;  205     0204  1 !
;  206     0205  1 FORWARD ROUTINE
;  207     0206  1      main,                                        ! Main entry
;  208     0207  1      scan_success          : NOVALUE,             ! Action routine for file found
;  209     0208  1      scan_failure          : NOVALUE,             !      "       "     file error
;  210     0209  1      exclude_file,                                ! Check exclude list
;  211     0210  1      validate_exclude_file : NOVALUE,             ! Verify that filename is OK
;  212     0211  1      locate_filename,                             ! Find filename field in name
;  213     0212  1      locate_version,                              ! Find version in filename
;  214     0213  1      match_file,                                  ! Compare filename strings
;  215     0214  1      search_one_file       : NOVALUE,             ! Open and search a file
;  216     0215  1      get_next_record       : JSB_GETNEXREC,       ! Read a record from the file
;  217     0216  1      advance_window        : NOVALUE,             ! Scroll the viewing window
;  218     0217  1      std_search_file       : NOVALUE,             ! Standard string matches
;  219     0218  1      std_match             : JSB_STDMAT,          ! See if search string present
;  220     0219  1      file_error,                                  ! Signal file I/O error
;  221     0220  1      upcase                : NOVALUE JSB_UPCASE,  ! Convert string to uppercase
;  222     0221  1      format_n_put          : NOVALUE JSB_FORPUT,  ! Format and put to output
;  223     0222  1      format_action_text    : JSB_FORACTTEX,       ! /FORMAT=TEXT routine
;  224     0223  1      format_action_dump    : JSB_FORACTDUM,       ! /FORMAT=DUMP & NONULLS
;  225     0224  1      srh$put_output        : NOVALUE JSB_PUTOUT;  ! Put record to output file
;  226     0225  1
;  227     0226  1 !
;  228     0227  1 ! External references
;  229     0228  1 !
;  230     0229  1 EXTERNAL ROUTINE
;  231     0230  1      lib$file_scan : addressing_mode(general),    ! Wildcard file searches
;  232     0231  1      lib$lookup_key: addressing_mode(general),    ! Keyword table lookup
;  233     0232  1      lib$put_output: addressing_mode(general),    ! Put stat to sys$output
;  234     0233  1      cli$get_value : addressing_mode(general),    ! Get qualifier value
;  235     0234  1      cli$present   : addressing_mode(general),    ! Get qualifier presence
;  236     0235  1      ots$cvt_ti_l  : addressing_mode(general),    ! ASCII decimal to longword
;  237     0236  1      str$get1_dx   : addressing_mode(general),    ! Allocate Dynamic String
;  238     0237  1      str$copy_dx   : addressing_mode(general),    ! Copy any class string
;  239     0238  1      lib$get_vm    : addressing_mode(general);    ! Gets memory
;  240     0239  1
;  241     0240  1 !
;  242     0241  1 ! Equated symbols
;  243     0242  1 !
;  244     0243  1 LITERAL
;  245     0244  1      true                  = 1,                   ! Routine value of TRUE
;  246     0245  1      false                 = 0,                   ! Routine value of FALSE
;  247     0246  1      max_infiles           = 136,                 ! Max files in SOURCE list
;  248     0247  1      max_srhstr            = 136,                 ! Max strings in STRING list
;  249     0248  1      max_exclude           = 136,                 ! Max files in EXCLUDE list
;  250     0249  1      io_buff_sz            = 2048,                ! Input buffer size
;  251     0250  1      form_text             = 0,                   ! Default format
;  252     0251  1      form_passall          = 1,                   ! Pass all characters
;  253     0252  1      form_dump             = 2,                   ! Reformat, flag 8 bit chars
```

```
;  254          0253  1     form_nonulls        = 3;                    ! Like DUMP, but no nulls
;  255          0254  1
;  256          0255  1 EXTERNAL LITERAL                    ! Facility-specific message definitions
;  257          0256  1     srh$_badexcl,
;  258          0257  1     srh$_badexclnam,
;  259          0258  1     srh$_badform,
;  260          0259  1     srh$_badmatch,
;  261          0260  1     srh$_matched,
;  262          0261  1     srh$_nomatch,
;  263          0262  1     srh$_nofile,
;  264          0263  1     srh$_nomatches,
;  265          0264  1     srh$_nostring,
;  266          0265  1     srh$_nullfile,
;  267          0266  1     srh$_rfaerr,
;  268          0267  1     srh$_truncate,
;  269          0268  1     srh$_wdw_maxprev,
;  270          0269  1     srh$_wdw_maxprm;
;  271          0270  1
;  272          0271  1                                     ! Shared message definitions
;  273        P 0272  1 $SHR_MSGDEF
;  274        P 0273  1     (srh, 215, local,
;  275        P 0274  1         (badlogic, error),
;  276        P 0275  1         (badvalue, error),
;  277        P 0276  1         (insvirmem, error),
;  278        P 0277  1         (openin, warning),
;  279        P 0278  1         (openout, error),
;  280        P 0279  1         (readerr, warning),
;  281          0280  1         (writeerr, severe));
;  282          0281  1
;  283          0282  1
;  284          0283  1 !
;  285          0284  1 ! OWN storage, initialized and demand-zero variables are defined
;  286          0285  1 ! in separate groups so that the linker will give us a demand-zero
;  287          0286  1 ! section rather than storing a bunch of zeroes in the image file
;  288          0287  1 !
;  289          0288  1 OWN_DZRO
;  290          0289  1     exp_name    : VECTOR [nam$c_maxrss, BYTE],  ! RMS expanded name buffer
;  291          0290  1     res_name    : VECTOR [nam$c_maxrss, BYTE],  ! RMS result name buffer
;  292          0291  1     out_exp_nam : VECTOR [nam$c_maxrss, BYTE],  ! RMS expanded name buffer
;  293          0292  1     out_res_nam : VECTOR [nam$c_maxrss, BYTE];  ! RMS result name buffer
;  294          0293  1
;  295          0294  1 OWN_CRF
;  296          0295  1     string_desc : dyn_str_desc,                 ! Input search string desc
;  297          0296  1     mod_desc    : dyn_str_desc,                 ! Modifier string descriptor
;  298          0297  1     tmp_desc    : stat_str_desc(0, 0),          ! Kosher desc for misc fcns
;  299          0298  1
;  300        P 0299  1     nam_block   : $NAM (                        ! File name block
;  301        P 0300  1                         RSA = res_name,         !  Result name addr
;  302        P 0301  1                         RSS = nam$c_maxrss,     !  Result name size
;  303        P 0302  1                         ESA = exp_name,         !  Expanded name addr
;  304          0303  1                         ESS = nam$c_maxrss),    !  Expanded name size
;  305        P 0304  1     fab         : $FAB(                         ! Input file FAB
;  306        P 0305  1                         FOP = NAM,              !  Open by name block
;  307        P 0306  1                         SHR = (GET,PUT,UPI),    !  Allow other readers/writers
;  308        P 0307  1                         DNA = UPLIT BYTE('*.*'),!  Default file spec.
;  309        P 0308  1                         DNS = 3,                !   (DNM doesn't like *'s)
;  310          0309  1                         NAM = nam_block),       !  Name block
```

```
;  311        P 0310  1     rab          : $RAB(                          ! Input file RAB
;  312        P 0311  1                         MBF = 3,                  !  Multi-buffer count
;  313        P 0312  1                         RAC = SEQ,                !  Will also use RFA as needed
;  314        P 0313  1                         ROP = (RAH,LOC),          !  Read-ahead, locate mode
;  315        P 0314  1                         USZ = io_buff_sz,         !  User buffer size
;  316          0315  1                         FAB = fab),               !  FAB addr
;  317        P 0316  1     out_nam_blk  : $NAM (                         ! File name block
;  318        P 0317  1                         RSA = out_res_nam,        !  Result name addr
;  319        P 0318  1                         RSS = nam$c_maxrss,       !  Result name size
;  320        P 0319  1                         ESA = out_exp_nam,        !  Expanded name addr
;  321          0320  1                         ESS = nam$c_maxrss),      !  Expanded name size
;  322        P 0321  1     outfab       : $FAB(                          ! Output file FAB
;  323        P 0322  1                         RAT = CR,                 !  Record attributes
;  324        P 0323  1                         DNA = UPLIT BYTE('SEARCH.LIS'), !  Default file spec.
;  325        P 0324  1                         DNS = 10,
;  326          0325  1                         NAM = out_nam_blk),       !  Name block
;  327        P 0326  1     outrab       : $RAB(                          ! Output file RAB
;  328        P 0327  1                         ROP = WBH,                !  Write-behind
;  329          0328  1                         FAB = outfab),            !  FAB addr
;  330          0329  1
;  331          0330  1     ! Build a keyword table for the FORMAT qualifier
;  332          0331  1     !
;  333          0332  1     format_keytbl       : VECTOR [9, LONG] PRESET (
;  334          0333  1                   [0] = 8,                              ! Entries + values
;  335          0334  1                   [1] = UPLIT BYTE (%ASCIC 'TEXT'),     ! String for TEXT
;  336          0335  1                   [2] = form_text,                      ! Value for TEXT
;  337          0336  1                   [3] = UPLIT BYTE (%ASCIC 'PASSALL'),  ! String
;  338          0337  1                   [4] = form_passall,                   ! Value
;  339          0338  1                   [5] = UPLIT BYTE (%ASCIC 'DUMP'),     ! String
;  340          0339  1                   [6] = form_dump,                              ! Value
;  341          0340  1                   [7] = UPLIT BYTE (%ASCIC 'NONULLS'),  ! String
;  342          0341  1                   [8] = form_nonulls),                  ! Value
;  343          0342  1     !
;  344          0343  1     ! Build a keyword table for the MATCH qualifier
;  345          0344  1     !
;  346          0345  1     match_keytbl        : VECTOR [9, LONG] PRESET (
;  347          0346  1                   [0] = 8,                              ! Entries + values
;  348          0347  1                   [1] = UPLIT BYTE (%ASCIC 'OR'),       ! String for OR
;  349          0348  1                   [2] = 0,                              ! Value for OR
;  350          0349  1                   [3] = UPLIT BYTE (%ASCIC 'NOR'),      ! String
;  351          0350  1                   [4] = 1,                              ! Value
;  352          0351  1                   [5] = UPLIT BYTE (%ASCIC 'AND'),      ! String
;  353          0352  1                   [6] = 2,                              ! Value
;  354          0353  1                   [7] = UPLIT BYTE (%ASCIC 'NAND'),     ! String
;  355          0354  1                   [8] = 3);                             ! Value
;  356          0355  1
;  357          0356  1 !
;  358          0357  1 ! Now define all the non-initialized variables
;  359          0358  1 !
;  360          0359  1 OWN_DZRO
;  361          0360  1     wdw_pre_recs,                                 ! Previous records requested
;  362          0361  1     wdw_sub_recs,                                 ! Subsequent records requested
;  363          0362  1     wdw_pre_cntr,                                 ! Previous records available
;  364          0363  1     wdw_sub_cntr,                                 ! Subsequent records to list
;  365          0364  1     wdw_pre_buff,                                 ! Current io_buff index
;  366          0365  1     cnt_infile,                                   ! Actual # of input files
;  367          0366  1     cnt_srhstr,                                   ! Actual # of search strings
```

```
;  368          0367  1      cnt_exclude,                            ! Actual # of exclude files
;  369          0368  1      max_rec       : VOLATILE,               ! Actual output max rec size
;  370          0369  1      format,                                 ! Code for output format
;  371          0370  1      fil_totmat,                             ! Total matches in one file
;  372          0371  1      fil_linenum,                            ! Total lines in one file
;  373          0372  1      out_file_open : BYTE,                   ! An output file is open
;  374          0373  1      out_printing  : BYTE,                   ! Matched records to output
;  375          0374  1      wdw_previous  : BYTE,                   ! RFA windowing active
;  376          0375  1      wdw_zero      : BYTE,                   ! We have /WINDOW=0
;  377          0376  1      wdw_frame     : BYTE,                   ! Frame discontiguous windows
;  378          0377  1      wdw_ovfl      : BYTE,                   ! Window discontiguity flag
;  379          0378  1      qua_exact     : BYTE,                   ! Exact case match flag
;  380          0379  1      qua_log       : BYTE,                   ! Log filenames searched
;  381          0380  1      qua_numbers   : BYTE,                   ! Output line number
;  382          0381  1      qua_heading   : BYTE,                   ! Display names of files
;  383          0382  1      new_file      : BYTE,                   ! Just opened a new file
;  384          0383  1      found_file    : BYTE,                   ! Found at least one file flag
;  385          0384  1      found_any     : BYTE,                   ! Any matches found boolean
;  386          0385  1      fil_found     : BYTE,                   ! Match found in this file
;  387          0386  1      mat_and       : BYTE,                   ! Match is AND if true, OR if false
;  388          0387  1      mat_negate    : BYTE,                   ! Turns above to NAND and NOR
;  389          0388  1
;  390          0389  1      rfa_vec       : REF bblockvector [,rab$s_rfa],! RFA's for previous records
;  391          0390  1
;  392          0391  1      output_buff : REF VECTOR [,BYTE],       ! For reformatting
;  393          0392  1
;  394          0393  1      infile_desc :                           ! Array of input filenames
;  395          0394  1                    bblockvector [max_infiles, dsc$k_d_bln],
;  396          0395  1      srhstr_desc :                           ! Array of search strings
;  397          0396  1                    bblockvector [max_srhstr,  dsc$k_d_bln],
;  398          0397  1      exclude_desc :                          ! Array of exclude filenames
;  399          0398  1                    bblockvector [max_exclude, dsc$k_d_bln];
;  400          0399  1
;  401        L 0400  1  %IF switch_statistics
;  402          0401  1  %THEN
;  403          0402  1      OWN_DZRO
;  404          0403  1          !
;  405          0404  1          ! define the block for the GETJPI for statistics
;  406          0405  1          !
;  407          0406  1          qua_statistics,                 ! Give statistics
;  408          0407  1          stat_totchr,                            ! Total characters in all files
;  409          0408  1          stat_totrec,                            ! Total records in all files
;  410          0409  1          stat_totfil,                            ! Total files searched
;  411          0410  1          stat_totmat,                            ! Total matches found
;  412          0411  1          stat_totput,                            ! Total records put to output
;  413          0412  1          stat_jpi_buffio,                        ! $GETJPI deposits them here
;  414          0413  1          stat_jpi_cputim,
;  415          0414  1          stat_jpi_dirio,
;  416          0415  1          stat_jpi_pagefIts,
;  417          0416  1          stat_buffio,                            ! Saved from startup getjpi
;  418          0417  1          stat_cputim,
;  419          0418  1          stat_dirio,
;  420          0419  1          stat_pagefIts,
;  421          0420  1          stat_starttime : VECTOR [2, LONG];
;  422          0421  1
;  423          0422  1      OWN_CRF
;  424          0423  1          stat_jpi_block : VECTOR [13, LONG] INITIAL (
```

```
;  425          0424  1                          (JPI$_BUFIO^16 OR 4),   stat_jpi_bufio, 0,
;  426          0425  1                          (JPI$_CPUTIM^16 OR 4),  stat_jpi_cputim, 0,
;  427          0426  1                          (JPI$_DIRIO^16 OR 4),   stat_jpi_dirio, 0,
;  428          0427  1                          (JPI$_PAGEFLTS^16 OR 4),stat_jpi_pagefIts, 0,
;  429          0428  1                          0);
;  430          0429  1 %FI ! switch_statistics
;  431          0430  1
;  432          0431  1 BIND
;  433          0432  1     stars_30 = UPLIT BYTE ('******************************'),
;  434          0433  1     null_str = %ASCID '';                       ! The null string
```

```
;  436     0434  1 ROUTINE main =
;  437     0435  1
;  438     0436  1 !++
;  439     0437  1 ! Functional description
;  440     0438  1 !
;  441     0439  1 !       This routine is the transfer point and central loop for the
;  442     0440  1 !       utility.  It retrieves filename(s), search string(s) and qualifiers
;  443     0441  1 !       from DCL, and looks up the files and calls the search routines.
;  444     0442  1 !
;  445     0443  1 ! Calling sequence
;  446     0444  1 !
;  447     0445  1 !       main ()
;  448     0446  1 !
;  449     0447  1 ! Input parameters
;  450     0448  1 !
;  451     0449  1 !       none
;  452     0450  1 !
;  453     0451  1 ! Routine value
;  454     0452  1 !
;  455     0453  1 !       Most severe error encountered during processing or SS$_NORMAL
;  456     0454  1 !
;  457     0455  1 !--
;  458     0456  2 BEGIN
;  459     0457  2
;  460     0458  2 LOCAL
;  461     0459  2     !
;  462     0460  2     ! Obtain memory for the input buffer and the output buffer
;  463     0461  2     ! Worst case size of output buffer is 6*input + 8 (when full
;  464     0462  2     ! length input is all -1's, /FORM=DUMP/NUMBERS)
;  465     0463  2     !
;  466     0464  2     out_buffer  : BLOCK [io_buff_sz*6+8, BYTE], ! For reformatting
;  467     0465  2     inp_buffer  : BLOCK [io_buff_sz, BYTE];     ! Input buffer
;  468     0466  2
;  469     0467  2 LOCAL
;  470     0468  2     wdw_1,                          ! Flag that first window parm
;  471     0469  2     wdw_2,                          ! Flag that 2 window parameters
;  472     0470  2     window_1,                       ! Value of window parm 1
;  473     0471  2     window_2,                       ! Value of window parm 2
;  474     0472  2     status,                         ! Error status
;  475     0473  2     keyval,                         ! Will receive keyword value
;  476     0474  2     scan_context;                   ! Sticky context argument for LIB$FILE_SCAN
;  477     0475  2
;  478     0476  2 BIND
;  479     0477  2     window_str = %ASCID 'WINDOW',               ! Qualifier keywords
;  480     0478  2     string_str = %ASCID 'STRING',
;  481     0479  2     source_str = %ASCID 'SOURCE';
;  482     0480  2
;  483     0481  2 !
;  484     0482  2 ! Get initial run statistics
;  485     0483  2 !
;  486   L 0484  2 %IF switch_statistics
;  487     0485  2 %THEN
;  488     0486  2     IF qua_statistics = cli$present(%ASCID 'STATISTICS')
;  489     0487  2     THEN
;  490     0488  3         BEGIN
;  491     0489  3
;  492     0490  4         IF NOT (status = $GETJPI (ITMLST=stat_jpi_block))
```

```
;  493     0491  3          THEN
;  494     0492  3              RETURN .status;
;  495     0493  4          IF NOT (status = $GETTIM (TIMADR=stat_starttime))
;  496     0494  3          THEN
;  497     0495  3              RETURN .status;
;  498     0496  3
;  499     0497  3          stat_bufio    = .stat_jpi_bufio;        ! copy from the JPI return variables
;  500     0498  3          stat_cputim   = .stat_jpi_cputim;
;  501     0499  3          stat_dirio    = .stat_jpi_dirio;
;  502     0500  3          stat_pageflts = .stat_jpi_pageflts;
;  503     0501  2          END;
;  504     0502  2 %FI
;  505     0503  2 !
;  506     0504  2 ! Set some pointers to the local data for this procedure
;  507     0505  2 !
;  508     0506  2
;  509     0507  2 rab [rab$l_ubf] = inp_buffer;                  ! Address of user buffer
;  510     0508  2 output_buff = out_buffer;                      ! Buffer for reformatting
;  511     0509  2
;  512     0510  2 !
;  513     0511  2 ! Get the simple qualifiers
;  514     0512  2 !
;  515     0513  2 qua_exact      = cli$present(%ASCID 'EXACT');   ! Want lower case matches?
;  516     0514  2 qua_log        = cli$present(%ASCID 'LOG');     ! Show filenames?
;  517     0515  2 qua_numbers    = cli$present(%ASCID 'NUMBERS'); ! Want line numbers?
;  518     0516  2 qua_heading    = cli$present(%ASCID 'HEADING'); ! Print file headings?
;  519     0517  2
;  520     0518  2 !
;  521     0519  2 ! Get the window size parameters
;  522     0520  2 !
;  523     0521  2 IF wdw_1 = cli$get_value(window_str, mod_desc)
;  524     0522  2 THEN
;  525     0523  3          BEGIN
;  526     0524  3          IF NOT ots$cvt_ti_l(mod_desc,window_1)
;  527     0525  3          THEN
;  528     0526  4              BEGIN
;  529     0527  4              SIGNAL (srh$_badvalue, 1, mod_desc);
;  530     0528  4              RETURN (srh$_badvalue OR %X'10000000');  ! Inhibit DCL PUTMSG
;  531     0529  3              END;
;  532     0530  3          IF .window_1 LSS 0                ! BLISS will optimize these together
;  533     0531  3          THEN
;  534     0532  4              BEGIN
;  535     0533  4              SIGNAL (srh$_badvalue, 1, mod_desc);
;  536     0534  4              RETURN (srh$_badvalue OR %X'10000000');
;  537     0535  3              END;
;  538     0536  2          END;
;  539     0537  2
;  540     0538  2 ! Try to get the second window parameter
;  541     0539  2 !
;  542     0540  2 IF wdw_2 = cli$get_value(window_str, mod_desc)
;  543     0541  2 THEN
;  544     0542  3          BEGIN
;  545     0543  3          IF NOT ots$cvt_ti_l(mod_desc, window_2)
;  546     0544  3          THEN
;  547     0545  4              BEGIN
;  548     0546  4              SIGNAL (srh$_badvalue, 1, mod_desc);
;  549     0547  4              RETURN (srh$_badvalue OR %X'10000000');
```

```
;  550     0548  3              END;
;  551     0549  3          IF .window_2 LSS 0              ! BLISS will optimize these together
;  552     0550  3          THEN
;  553     0551  4              BEGIN
;  554     0552  4              SIGNAL (srh$_badvalue, 1, mod_desc);
;  555     0553  4              RETURN (srh$_badvalue OR %X'10000000');
;  556     0554  3              END;
;  557     0555  2          END;
;  558     0556  2
;  559     0557  2 IF cli$get_value(window_str, mod_desc) ! Make sure no more in list.
;  560     0558  2 THEN
;  561     0559  2     RETURN srh$_wdw_maxprm;
;  562     0560  2
;  563     0561  2 ! Set the appropriate values for previous recs and subsequent records.  We
;  564     0562  2 ! assume that WDW_PRE_RECS and WDW_SUB_RECS are initially zero.
;  565     0563  2 !
;  566     0564  2 IF .wdw_1                               ! If /NOWINDOW we have nothing to do.
;  567     0565  2 THEN
;  568     0566  3     BEGIN
;  569     0567  3     IF .wdw_2                           ! We have /WINDOW=(pre,sub) format.
;  570     0568  3     THEN
;  571     0569  4         BEGIN
;  572     0570  4         wdw_pre_recs = .window_1;
;  573     0571  4         wdw_sub_recs = .window_2;
;  574     0572  4         END
;  575     0573  3     ELSE                                ! We have /WINDOW=total format
;  576     0574  4         BEGIN
;  577     0575  4         IF .window_1 EQL 0
;  578     0576  4         THEN
;  579     0577  4             wdw_zero = 1                ! Set flag for /WINDOW=0 option
;  580     0578  4         ELSE
;  581     0579  5             BEGIN
;  582     0580  5             LOCAL                       ! Help BLISS optimize
;  583     0581  5                 tmp;
;  584     0582  5             tmp = .window_1 -1; ! Count the matched line as one of them
;  585     0583  5             wdw_pre_recs = .tmp / 2;
;  586     0584  5             wdw_sub_recs = .tmp - .wdw_pre_recs;
;  587     0585  4             END;
;  588     0586  3         END;
;  589     0587  2     END;
;  590     0588  2
;  591     0589  3 IF      (.wdw_pre_recs GTR 0)
;  592     0590  3     OR  (.wdw_sub_recs GTR 0)
;  593     0591  2 THEN
;  594     0592  2     wdw_frame = true;                   ! Don't say "***************" for /WINDOW=1
;  595     0593  2
;  596     0594  2 IF cli$present(%ASCID 'REMAINING')      ! Do we want match to end of file?
;  597     0595  2 THEN
;  598     0596  2     wdw_sub_recs = 2^31 - 1;            ! Set to something akin to infinity
;  599     0597  2
;  600     0598  2 !
;  601     0599  2 ! Open the output file
;  602     0600  2 !
;  603     0601  2 IF out_file_open = cli$get_value(%ASCID 'OUTPUT', mod_desc)
;  604     0602  2 THEN
;  605     0603  3         BEGIN
;  606     0604  3         outfab [fab$b_fns] = .mod_desc [dsc$w_length];
```

```
;  607     0605  3          outfab [fab$l_fna] = .mod_desc [dsc$a_pointer];
;  608     0606  4          IF NOT (status = $CREATE(FAB = outfab))
;  609     0607  3          THEN
;  610     0608  3              RETURN file_error (srh$_openout,.status,outfab,.outfab [fab$l_stv]);
;  611     0609  4          IF NOT (status = $CONNECT(RAB = outrab))
;  612     0610  3          THEN
;  613     0611  3              RETURN file_error (srh$_openout,.status,outfab,.outrab [rab$l_stv]);
;  614     0612  3          max_rec = .outfab [fab$w_mrs];
;  615     0613  3          IF .max_rec EQL 0 THEN max_rec = 2^15-1;
;  616     0614  2          END;
;  617     0615  2
;  618     0616  2 out_printing = (NOT .wdw_zero) AND .out_file_open; ! Do we print matched lines?
;  619     0617  2
;  620     0618  2 !
;  621     0619  2 ! Get the /MATCH qualifier value, set the bit vector
;  622     0620  2 !
;  623     0621  2 cli$get_value(%ASCID 'MATCH', string_desc);      ! Get the string
;  624     0622  2 upcase(.string_desc [dsc$w_length], .string_desc [dsc$a_pointer],
;  625     0623  2                                    .string_desc [dsc$a_pointer]);
;  626     0624  3 IF NOT (status = lib$lookup_key (string_desc, match_keytbl, keyval))
;  627     0625  2 THEN
;  628     0626  2     SIGNAL_STOP (srh$_badmatch, 0, .status, 1, string_desc);
;  629     0627  2
;  630     0628  2 CASE .keyval FROM 0 TO 3 OF
;  631     0629  2     SET
;  632     0630  2         [0] :   ;                       ! 'OR' - No action, both zero already
;  633     0631  2         [1] :   mat_negate = true;      ! 'NOR' - mat_and is already zero
;  634     0632  2         [2] :   mat_and = true;         ! 'AND' - mat_negate is already zero
;  635     0633  3         [3] :   BEGIN                   ! 'NAND' - We have to set them both
;  636     0634  3                 mat_and = true;
;  637     0635  3                 mat_negate = true;
;  638     0636  2                 END;
;  639     0637  2     TES;
;  640     0638  2
;  641     0639  2 !
;  642     0640  2 ! If there are no previous records we can set for sequential only
;  643     0641  2 !
;  644     0642  2
;  645     0643  2 IF .wdw_pre_recs EQL 0
;  646     0644  2 THEN                                                                    2A
;  647     0645  2     fab [fab$v_sqo] = true                                              2A
;  648     0646  2 ELSE
;  649     0647  3     BEGIN
;  650     0648  3     wdw_previous = true;
;  651     0649  3     !
;  652     0650  3     ! Obtain memory for the vector of RFA's
;  653     0651  3     !
;  654     0652  4     IF NOT (status = lib$get_vm(%REF((.wdw_pre_recs + 1)*rab$s_rfa), rfa_vec))
;  655     0653  3     THEN
;  656     0654  3         SIGNAL_STOP (srh$_wdw_maxprev, 0, .status);     ! Buffer max exceeded.
;  657     0655  2     END;
;  658     0656  2
;  659     0657  2 !
;  660     0658  2 ! Get the /FORMAT qualifier value, set the variable
;  661     0659  2 !
;  662     0660  2 cli$get_value(%ASCID 'FORMAT', string_desc);    ! Get the string
;  663     0661  2 upcase(.string_desc [dsc$w_length], .string_desc [dsc$a_pointer],
```

```
;  664     0662  2                                        .string_desc [dsc$a_pointer]);
;  665     0663  3 IF NOT (status = lib$lookup_key (string_desc, format_keytbl, format))
;  666     0664  2 THEN
;  667     0665  2     SIGNAL_STOP (srh$_badform, 0, .status, 1, string_desc);
;  668     0666  2
;  669     0667  2 !
;  670     0668  2 ! DCL parameter P1, SOURCE
;  671     0669  2 !
;  672     0670  2 ! Get all the input file names so that we will know if a list is specified.
;  673     0671  2 ! Max_infiles is large enough so that is is IMPOSSIBLE to overflow
;  674     0672  2 !
;  675     0673  2 WHILE .cnt_infile LSS max_infiles
;  676     0674  2 DO
;  677     0675  3     BEGIN
;  678     0676  3     infile_desc [.cnt_infile, dsc_long1] = dsc_dynam;   ! Init to dynamic desc
;  679     0677  4     IF NOT (status = cli$get_value(source_str,
;  680     0678  4                                    infile_desc [.cnt_infile, dsc$w_length]) )
;  681     0679  3     THEN
;  682     0680  3         EXITLOOP;
;  683     0681  3     cnt_infile = .cnt_infile+1;
;  684     0682  2     END;
;  685     0683  2
;  686     0684  2 !
;  687     0685  2 ! The command requires SOURCE, but just in case someone mungs the CLD file
;  688     0686  2 !
;  689     0687  2 IF .cnt_infile LEQ 0
;  690     0688  2 THEN
;  691     0689  2     RETURN srh$_nofile;
;  692     0690  2
;  693     0691  2 !
;  694     0692  2 ! DCL parameter P2, STRING
;  695     0693  2 !
;  696     0694  2 ! Get the all the search string entries
;  697     0695  2 ! Max_srhstr is large enough so that is is IMPOSSIBLE to overflow
;  698     0696  2 !
;  699     0697  2 WHILE .cnt_srhstr LSS max_srhstr
;  700     0698  2 DO
;  701     0699  3     BEGIN
;  702     0700  4     IF NOT (status = cli$get_value(string_str, string_desc))
;  703     0701  3     THEN
;  704     0702  3         EXITLOOP;
;  705     0703  3         !
;  706     0704  3         ! Copy or upcase the search string
;  707     0705  3         !
;  708     0706  3     srhstr_desc [.cnt_srhstr, dsc_long1] = dsc_dynam;   ! Init to dynamic desc
;  709     0707  3
;  710     0708  3     IF .qua_exact THEN                  ! Copy to descriptor array
;  711     0709  3         str$copy_dx (srhstr_desc [.cnt_srhstr,dsc$w_length],
;  712     0710  3                      string_desc [dsc$w_length])
;  713     0711  3     ELSE
;  714     0712  4         BEGIN                           ! Upcase into the descriptor array
;  715     0713  4         str$get1_dx (string_desc [dsc$w_length], ! Allocate Destination String
;  716     0714  4                      srhstr_desc [.cnt_srhstr,dsc$w_length]);
;  717     0715  4         upcase(                                 ! Uppercase search string
;  718     0716  4                .srhstr_desc [.cnt_srhstr, dsc$w_length],   !  Length of source
;  719     0717  4                .string_desc [dsc$a_pointer],               !  Source addr
;  720     0718  4                .srhstr_desc [.cnt_srhstr, dsc$a_pointer]); !  Destination addr
```

```
;  721     0719  3          END;
;  722     0720  3
;  723     0721  3      cnt_srhstr = .cnt_srhstr+1;
;  724     0722  2      END;
;  725     0723  2
;  726     0724  2  !
;  727     0725  2  ! The command requires STRING, but just in case someone changes the CLD file
;  728     0726  2  !
;  729     0727  2  IF .cnt_srhstr LEQ 0
;  730     0728  2  THEN
;  731     0729  2      RETURN srh$_nostring;
;  732     0730  2
;  733     0731  2  !
;  734     0732  2  ! DCL qualifier /EXCLUDE
;  735     0733  2  !
;  736     0734  2  ! Get the all the exclude filenames
;  737     0735  2  ! Max_exclude is large enough so that is is IMPOSSIBLE to overflow
;  738     0736  2  !
;  739     0737  2  WHILE .cnt_exclude LSS max_exclude
;  740     0738  2  DO
;  741     0739  3      BEGIN
;  742     0740  3      LOCAL
;  743     0741  3          buf : VECTOR [nam$c_maxrss, BYTE],
;  744     0742  3          inter_desc : BLOCK [8, BYTE],          ! Intermediate name
;  745     0743  3          result_desc : BLOCK [8, BYTE];
;  746     0744  3
;  747     0745  4      IF NOT (status = cli$get_value(%ASCID 'EXCLUDE', string_desc))
;  748     0746  3      THEN
;  749     0747  3          EXITLOOP;
;  750     0748  3
;  751     0749  3      ! Don't let device and directory sneak in with a logical name
;  752     0750  3      !
;  753     0751  3      CH$MOVE (8, string_desc, inter_desc);       ! Start with the given name
;  754     0752  3      CH$MOVE (4, tmp_desc, result_desc);         ! Static string desc info
;  755     0753  3      DO
;  756     0754  4          BEGIN
;  757     0755  4          result_desc [dsc$w_length] = nam$c_maxrss;
;  758     0756  4          result_desc [dsc$a_pointer] = buf;
;  759     0757  4          status = $trnlog (lognam = inter_desc, rsllen = result_desc, rslbuf = result_desc);
;  760     0758  4          IF NOT .status
;  761     0759  4          THEN
;  762     0760  4              SIGNAL_STOP (.status);
;  763     0761  4          CH$MOVE (8, result_desc, inter_desc);   ! Ready to translate again
;  764     0762  4          END
;  765     0763  3      UNTIL .status EQL ss$_notran;
;  766     0764  3
;  767     0765  3      !
;  768     0766  3      ! Make sure that the filename has no device or directory.  Allow for
;  769     0767  3      ! alternate directory syntax of <dir>
;  770     0768  3      !
;  771     0769  3      IF     CH$FIND_CH(.result_desc [dsc$w_length],
;  772     0770  3                       .result_desc [dsc$a_pointer], ':') NEQ 0
;  773     0771  3          OR CH$FIND_CH(.result_desc [dsc$w_length],
;  774     0772  3                       .result_desc [dsc$a_pointer], ']') NEQ 0
;  775     0773  3          OR CH$FIND_CH(.result_desc [dsc$w_length],
;  776     0774  3                       .result_desc [dsc$a_pointer], '>') NEQ 0
;  777     0775  3      THEN
```

```
;  778          0776  3          SIGNAL_STOP (srh$_badexcl,1,result_desc);       ! Scream and shout
;  779          0777  3
;  780          0778  3      !
;  781          0779  3      ! Copy, upcase and validate the exclude filename
;  782          0780  3      !
;  783          0781  3      exclude_desc [.cnt_exclude, dsc_long1] = dsc_dynam; ! Init to dynamic desc
;  784          0782  3      validate_exclude_file (result_desc,exclude_desc [.cnt_exclude, dsc$w_length]);
;  785          0783  3      cnt_exclude = .cnt_exclude+1;
;  786          0784  2      END;
```

```
;  788          0785  2 !
;  789          0786  2 ! Search the input files
;  790          0787  2 !
;  791          0788  2 found_file = false;                          ! Nothing found yet
;  792          0789  2 found_any = false;
;  793          0790  2 scan_context = 0;                           ! Sticky context argument must be zero
;  794          0791  2 INCR ifx FROM 0 TO .cnt_infile-1                    ! Try all of out
;  795          0792  2 DO                                                  ! file spec.
;  796          0793  3         BEGIN
;  797          0794  3         fab [fab$l_fna] = .infile_desc [.ifx, dsc$a_pointer];    ! Set name addr
;  798          0795  3         fab [fab$b_fns] = .infile_desc [.ifx, dsc$w_length];     ! Set name size
;  799          0796  3         lib$file_scan (fab,                                      ! Do full wildcard scan
;  800          0797  3                        scan_success,                             ! Go here on success
;  801          0798  3                        scan_failure,                             ! Go here on failure
;  802          0799  3                        scan_context);                            ! Sticky context argument
;  803          0800  3         fab [fab$l_dna] = 0;                                     ! Allow sticky type
;  804          0801  2         END;
```

```
;  806        L 0802  2 %IF switch_statistics
;  807          0803  2 %THEN
;  808          0804  2 !
;  809          0805  2 ! Display the statistics if desired.
;  810          0806  2 !
;  811          0807  2 IF .qua_statistics
;  812          0808  2 THEN
;  813          0809  3     BEGIN
;  814          0810  3     BUILTIN
;  815          0811  3         SUBM;
;  816          0812  3     LOCAL
;  817          0813  3         quadtime : VECTOR [2, LONG],
;  818          0814  3         tim,
;  819          0815  3         tim_buf : VECTOR [16, BYTE],
;  820          0816  3         tim_desc : VECTOR [2, LONG],
;  821          0817  3         stat,
;  822          0818  3         out_buf : VECTOR [80,BYTE];
;  823          0819  3
;  824          0820  4     IF NOT (stat = $GETJPI (ITMLST=stat_jpi_block))
;  825          0821  3     THEN
;  826          0822  3         RETURN .stat;
;  827          0823  3
;  828          0824  3     tmp_desc [dsc$a_pointer] = out_buf;
;  829          0825  3     lib$put_output(null_str);
;  830          0826  3
;  831          0827  3     tmp_desc [dsc$w_length] = 80;
;  832        P 0828  4     IF NOT (stat = $FAO( %ASCID 'Files searched:        !10UL       Buffered I/O count:!10UL',
;  833          0829  4                 tmp_desc, tmp_desc, .stat_totfil, .stat_jpi_bufio-.stat_bufio))
;  834          0830  3     THEN
;  835          0831  3         RETURN .stat;   ! Signal the error
;  836          0832  3     lib$put_output(tmp_desc);
;  837          0833  3
;  838          0834  3     tmp_desc [dsc$w_length] = 80;
;  839        P 0835  4     IF NOT (stat = $FAO( %ASCID 'Records searched:      !10UL       Direct I/O count:  !10UL',
;  840          0836  4                 tmp_desc, tmp_desc, .stat_totrec, .stat_jpi_dirio-.stat_dirio))
;  841          0837  3     THEN
;  842          0838  3         RETURN .stat;   ! Signal the error
;  843          0839  3     lib$put_output(tmp_desc);
;  844          0840  3
;  845          0841  3     tmp_desc [dsc$w_length] = 80;
;  846        P 0842  4     IF NOT (stat = $FAO( %ASCID 'Characters searched:   !10UL       Page faults:       !10UL',
;  847          0843  4                 tmp_desc, tmp_desc, .stat_totchr, .stat_jpi_pagefIts-.stat_pagefIts))
;  848          0844  3     THEN
;  849          0845  3         RETURN .stat;   ! Signal the error
;  850          0846  3     lib$put_output(tmp_desc);
;  851          0847  3
;  852          0848  3     tmp_desc [dsc$w_length] = 80;
;  853          0849  3     tim = .stat_jpi_cputim-.stat_cputim;              ! get the elapsed CPU time
;  854        P 0850  4     IF NOT (stat = $FAO( %ASCID 'Records matched:       !10UL       Elapsed CPU time:!3UL !2ZL:!2ZL:!2ZL.!2Z
;  855        P 0851  4                 tmp_desc, tmp_desc, .stat_totmat,
;  856        P 0852  4                 (.tim/(100*60*60*24)),              ! days
;  857        P 0853  4                 (.tim/(100*60*60)) MOD 24,          ! hours
;  858        P 0854  4                 (.tim/(100*60)) MOD 60,             ! minutes
;  859        P 0855  4                 (.tim/100) MOD 60,                  ! seconds
;  860          0856  4                 .tim MOD 100))                      ! hundredths of seconds
;  861          0857  3     THEN
;  862          0858  3         RETURN .stat;   ! Signal the error
```

```
;  863          0859  3     lib$put_output(tmp_desc);
```

```
;  865          0860  3     tim_desc [0] = 16;
;  866          0861  3     tim_desc [1] = tim_buf;
;  867          0862  4     IF NOT (stat = $GETTIM (TIMADR=quadtime))
;  868          0863  3     THEN
;  869          0864  3         RETURN .stat;   ! Signal the error
;  870          0865  3     SUBM (2, quadtime, stat_starttime, quadtime);
;  871          0866  4     IF NOT (stat = $ASCTIM (TIMLEN=tim_desc, TIMBUF=tim_desc, TIMADR=quadtime))
;  872          0867  3     THEN
;  873          0868  3         RETURN .stat;   ! Signal the error
;  874          0869  3
;  875          0870  3     tmp_desc [dsc$w_length] = 80;
;  876        P 0871  4     IF NOT (stat = $FAO( %ASCID 'Lines printed:         !10UL       Elapsed time:   !AS',
;  877          0872  4                 tmp_desc, tmp_desc, .stat_totput, tim_desc))
;  878          0873  3     THEN
;  879          0874  3         RETURN .stat;   ! Signal the error
;  880          0875  3     lib$put_output(tmp_desc);
;  881          0876  3
;  882          0877  2     END;
;  883          0878  2 %FI ! switch_statistics
;  884          0879  2
;  885          0880  2 IF .out_file_open                       ! All through writing
;  886          0881  2 THEN
;  887          0882  2     $CLOSE(FAB = outfab);
;  888          0883  2
;  889          0884  2 status = ss$_normal;
;  890          0885  2 IF NOT .found_any                       ! If no matches were found
;  891          0886  2 THEN
;  892          0887  2         IF .found_file                  ! set non-success status.
;  893          0888  2         THEN
;  894          0889  2                 SIGNAL (status = srh$_nomatches)
;  895          0890  2         ELSE
;  896          0891  2                 status = srh$_nofile;
;  897          0892  2
;  898          0893  2 RETURN .status;
;  899          0894  1 END;


                                        .TITLE  SEARCH
                                        .IDENT  \V04-000\

                                        .PSECT  $OWN$_CRF,NOEXE,2

                                    00# 00000 STRING_DESC:
                                        .BYTE   0[2]                                    ;
                                 02 0E  00002          .BYTE   14, 2                    ;
                                        00004          .BLKB   4
                                    00# 00008 MOD_DESC:
                                        .BYTE   0[2]                                    ;
                                 02 0E  0000A          .BYTE   14, 2                    ;
                                        0000C          .BLKB   4
                                  0000  00010 TMP_DESC:
                                        .WORD   0                                       ;
                                 01 0E  00012          .BYTE   14, 1                    ;
                               00000000 00014          .LONG   0                        ;
                                    02  00018 NAM_BLOCK:
                                        .BYTE   2                                       ;
                                    60  00019          .BYTE   96                       ;
```

```
                FF  0001A          .BYTE   -1
                00  0001B          .BYTE   0
00000000'       0001C          .ADDRESS RES_NAME
                00  00020          .BYTE   0
                00  00021          .BYTE   0
                FF  00022          .BYTE   -1
                00  00023          .BYTE   0
00000000'       00024          .ADDRESS EXP_NAME
00000000        00028          .LONG   0
            0000#   0002C          .WORD   0[8]
            0000#   0003C          .WORD   0[3]
            0000#   00042          .WORD   0[3]
00000000        00048          .LONG   0
00000000        0004C          .LONG   0
                00  00050          .BYTE   0
                00  00051          .BYTE   0
                00  00052          .BYTE   0
                00  00053          .BYTE   0
                00  00054          .BYTE   0
                00  00055          .BYTE   0
                00# 00056          .BYTE   0[2]
00000000        00058          .LONG   0
00000000        0005C          .LONG   0
00000000        00060          .LONG   0
00000000        00064          .LONG   0
00000000        00068          .LONG   0
00000000        0006C          .LONG   0
00000000#       00070          .LONG   0[2]
                03  00078 FAB:    .BYTE   3
                50  00079          .BYTE   80
              0000  0007A          .WORD   0
01000000        0007C          .LONG   16777216
00000000        00080          .LONG   0
00000000        00084          .LONG   0
00000000        00088          .LONG   0
              0000  0008C          .WORD   0
                02  0008E          .BYTE   2
                43  0008F          .BYTE   67
00000000        00090          .LONG   0
                00  00094          .BYTE   0
                00  00095          .BYTE   0
                00  00096          .BYTE   0
                02  00097          .BYTE   2
00000000        00098          .LONG   0
00000000        0009C          .LONG   0
00000000'       000A0          .ADDRESS NAM_BLOCK
00000000        000A4          .LONG   0
00000000'       000A8          .ADDRESS P.AAA
                00  000AC          .BYTE   0
                03  000AD          .BYTE   3
              0000  000AE          .WORD   0
00000000        000B0          .LONG   0
              0000  000B4          .WORD   0
                00  000B6          .BYTE   0
                00  000B7          .BYTE   0
00000000        000B8          .LONG   0
00000000        000BC          .LONG   0
```

```
                0000  000C0          .WORD   0
                  00  000C2          .BYTE   0
                  00  000C3          .BYTE   0
            00000000  000C4          .LONG   0
                  01  000C8 RAB:     .BYTE   1
                  44  000C9          .BYTE   68
                0000  000CA          .WORD   0
            00010200  000CC          .LONG   66048
            00000000  000D0          .LONG   0
            00000000  000D4          .LONG   0
                0000# 000D8          .WORD   0[3]
                0000  000DE          .WORD   0
            00000000  000E0          .LONG   0
                0000  000E4          .WORD   0
                  00  000E6          .BYTE   0
                  00  000E7          .BYTE   0
                0800  000E8          .WORD   2048
                0000  000EA          .WORD   0
            00000000  000EC          .LONG   0
            00000000  000F0          .LONG   0
            00000000  000F4          .LONG   0
            00000000  000F8          .LONG   0
                  00  000FC          .BYTE   0
                  00  000FD          .BYTE   0
                  03  000FE          .BYTE   3
                  00  000FF          .BYTE   0
            00000000  00100          .LONG   0
            00000000' 00104          .ADDRESS FAB
            00000000  00108          .LONG   0
                  02  0010C OUT_NAM_BLK:
                                     .BYTE   2
                  60  0010D          .BYTE   96
                  FF  0010E          .BYTE   -1
                  00  0010F          .BYTE   0
            00000000' 00110          .ADDRESS OUT_RES_NAM
                  00  00114          .BYTE   0
                  00  00115          .BYTE   0
                  FF  00116          .BYTE   -1
                  00  00117          .BYTE   0
            00000000' 00118          .ADDRESS OUT_EXP_NAM
            00000000  0011C          .LONG   0
                0000# 00120          .WORD   0[8]
                0000# 00130          .WORD   0[3]
                0000# 00136          .WORD   0[3]
            00000000  0013C          .LONG   0
            00000000  00140          .LONG   0
                  00  00144          .BYTE   0
                  00  00145          .BYTE   0
                  00  00146          .BYTE   0
                  00  00147          .BYTE   0
                  00  00148          .BYTE   0
                  00  00149          .BYTE   0
                  00# 0014A          .BYTE   0[2]
            00000000  0014C          .LONG   0
            00000000  00150          .LONG   0
            00000000  00154          .LONG   0
            00000000  00158          .LONG   0
```

```
00000000  0015C            .LONG   0
00000000  00160            .LONG   0
00000000# 00164            .LONG   0[2]
       03  0016C OUTFAB:   .BYTE   3
       50  0016D            .BYTE   80
     0000  0016E            .WORD   0
00000000  00170            .LONG   0
00000000  00174            .LONG   0
00000000  00178            .LONG   0
00000000  0017C            .LONG   0
     0000  00180            .WORD   0
       02  00182            .BYTE   2
       00  00183            .BYTE   0
00000000  00184            .LONG   0
       00  00188            .BYTE   0
       00  00189            .BYTE   0
       02  0018A            .BYTE   2
       02  0018B            .BYTE   2
00000000  0018C            .LONG   0
00000000  00190            .LONG   0
00000000' 00194            .ADDRESS OUT_NAM_BLK
00000000  00198            .LONG   0
00000000' 0019C            .ADDRESS P.AAB
       00  001A0            .BYTE   0
       0A  001A1            .BYTE   10
     0000  001A2            .WORD   0
00000000  001A4            .LONG   0
     0000  001A8            .WORD   0
       00  001AA            .BYTE   0
       00  001AB            .BYTE   0
00000000  001AC            .LONG   0
00000000  001B0            .LONG   0
     0000  001B4            .WORD   0
       00  001B6            .BYTE   0
       00  001B7            .BYTE   0
00000000  001B8            .LONG   0
       01  001BC OUTRAB:   .BYTE   1
       44  001BD            .BYTE   68
     0000  001BE            .WORD   0
00000400  001C0            .LONG   1024
00000000  001C4            .LONG   0
00000000  001C8            .LONG   0
     0000# 001CC            .WORD   0[3]
     0000  001D2            .WORD   0
00000000  001D4            .LONG   0
     0000  001D8            .WORD   0
       00  001DA            .BYTE   0
       00  001DB            .BYTE   0
     0000  001DC            .WORD   0
     0000  001DE            .WORD   0
00000000  001E0            .LONG   0
00000000  001E4            .LONG   0
00000000  001E8            .LONG   0
00000000  001EC            .LONG   0
       00  001F0            .BYTE   0
       00  001F1            .BYTE   0
       00  001F2            .BYTE   0
```

```
                    00  001F3          .BYTE   0
00000000  001F4          .LONG   0
00000000' 001F8          .ADDRESS OUTFAB
00000000  001FC          .LONG   0
00000008  00200 FORMAT_KEYTBL:
                         .LONG   8
00000000' 00204          .ADDRESS P.AAC
00000000  00208          .LONG   0
00000000' 0020C          .ADDRESS P.AAD
00000001  00210          .LONG   1
00000000' 00214          .ADDRESS P.AAE
00000002  00218          .LONG   2
00000000' 0021C          .ADDRESS P.AAF
00000003  00220          .LONG   3
00000008  00224 MATCH_KEYTBL:
                         .LONG   8
00000000' 00228          .ADDRESS P.AAG
00000000  0022C          .LONG   0
00000000' 00230          .ADDRESS P.AAH
00000001  00234          .LONG   1
00000000' 00238          .ADDRESS P.AAI
00000002  0023C          .LONG   2
00000000' 00240          .ADDRESS P.AAJ
00000003  00244          .LONG   3
040C0004  00248 STAT_JPI_BLOCK:
                         .LONG   67895300
00000000' 0024C          .ADDRESS STAT_JPI_BUFIO
04070004  00000000 00250 .LONG   0, 67567620
00000000' 00258          .ADDRESS STAT_JPI_CPUTIM
040B0004  00000000 0025C .LONG   0, 67829764
00000000' 00264          .ADDRESS STAT_JPI_DIRIO
040A0004  00000000 00268 .LONG   0, 67764228
00000000' 00270          .ADDRESS STAT_JPI_PAGEFLTS
00000000  00000000 00274 .LONG   0, 0

                         .PSECT  $OWN$_DZRO,NOEXE,2

                    00000 EXP_NAME:
                         .BLKB   255
                    000FF .BLKB   1
                    00100 RES_NAME:
                         .BLKB   255
                    001FF .BLKB   1
                    00200 OUT_EXP_NAM:
                         .BLKB   255
                    002FF .BLKB   1
                    00300 OUT_RES_NAM:
                         .BLKB   255
                    003FF .BLKB   1
                    00400 WDW_PRE_RECS:
                         .BLKB   4
                    00404 WDW_SUB_RECS:
                         .BLKB   4
                    00408 WDW_PRE_CNTR:
                         .BLKB   4
                    0040C WDW_SUB_CNTR:
                         .BLKB   4
```

```
00410 WDW_PRE_BUFF:
                .BLKB   4
00414 CNT_INFILE:
                .BLKB   4
00418 CNT_SRHSTR:
                .BLKB   4
0041C CNT_EXCLUDE:
                .BLKB   4
00420 MAX_REC:  .BLKB   4
00424 FORMAT:   .BLKB   4
00428 FIL_TOTMAT:
                .BLKB   4
0042C FIL_LINENUM:
                .BLKB   4
00430 OUT_FILE_OPEN:
                .BLKB   1
00431 OUT_PRINTING:
                .BLKB   1
00432 WDW_PREVIOUS:
                .BLKB   1
00433 WDW_ZERO:
                .BLKB   1
00434 WDW_FRAME:
                .BLKB   1
00435 WDW_OVFL:
                .BLKB   1
00436 QUA_EXACT:
                .BLKB   1
00437 QUA_LOG:  .BLKB   1
00438 QUA_NUMBERS:
                .BLKB   1
00439 QUA_HEADING:
                .BLKB   1
0043A NEW_FILE:
                .BLKB   1
0043B FOUND_FILE:
                .BLKB   1
0043C FOUND_ANY:
                .BLKB   1
0043D FIL_FOUND:
                .BLKB   1
0043E MAT_AND:  .BLKB   1
0043F MAT_NEGATE:
                .BLKB   1
00440 RFA_VEC:  .BLKB   4
00444 OUTPUT_BUFF:
                .BLKB   4
00448 INFILE_DESC:
                .BLKB   1088
00888 SRHSTR_DESC:
                .BLKB   1088
00CC8 EXCLUDE_DESC:
                .BLKB   1088
01108 QUA_STATISTICS:
                .BLKB   4
0110C STAT_TOTCHR:
                .BLKB   4
```

```
                        01110 STAT_TOTREC:
                                .BLKB   4
                        01114 STAT_TOTFIL:
                                .BLKB   4
                        01118 STAT_TOTMAT:
                                .BLKB   4
                        0111C STAT_TOTPUT:
                                .BLKB   4
                        01120 STAT_JPI_BUFIO:
                                .BLKB   4
                        01124 STAT_JPI_CPUTIM:
                                .BLKB   4
                        01128 STAT_JPI_DIRIO:
                                .BLKB   4
                        0112C STAT_JPI_PAGEFLTS:
                                .BLKB   4
                        01130 STAT_BUFIO:
                                .BLKB   4
                        01134 STAT_CPUTIM:
                                .BLKB   4
                        01138 STAT_DIRIO:
                                .BLKB   4
                        0113C STAT_PAGEFLTS:
                                .BLKB   4
                        01140 STAT_STARTTIME:
                                .BLKB   8

                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                2A  2E  2A  00000 P.AAA:  .ASCII  \*.*\                                 ;
                                        53  49  4C  2E  48  43  52  41  45  53  00003 P.AAB:  .ASCII  \SEARCH.LIS\                          ;
                                                        54  58  45  54  04  0000D P.AAC:  .ASCII  <4>\TEXT\                          ;
                                                    4C  4C  41  53  53  41  50  07  00012 P.AAD:  .ASCII  <7>\PASSALL\                       ;
                                                        50  4D  55  44  04  0001A P.AAE:  .ASCII  <4>\DUMP\                          ;
                                                    53  4C  4C  55  4E  4F  4E  07  0001F P.AAF:  .ASCII  <7>\NONULLS\                       ;
                                                                    52  4F  02  00027 P.AAG:  .ASCII  <2>\OR\                        ;
                                                                52  4F  4E  03  0002A P.AAH:  .ASCII  <3>\NOR\                       ;
                                                                44  4E  41  03  0002E P.AAI:  .ASCII  <3>\AND\                       ;
                                                            44  4E  41  4E  04  00032 P.AAJ:  .ASCII  <4>\NAND\                      ;
2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  00037 P.AAK:  .ASCII  \**********************************\           ;
2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  00046
                                                                00055        .BLKB   3
                                                                00058 P.AAM:  .BLKB   0                                         ;
                                                        010E0000  00058 P.AAL:  .LONG   17694720                                ;
                                                        00000000' 0005C         .ADDRESS P.AAM                                  ;
                                                    00  00  57  4F  44  4E  49  57  00060 P.AAO:  .ASCII  \WINDOW\<0><0>             ;
                                                        010E0006  00068 P.AAN:  .LONG   17694726                                ;
                                                        00000000' 0006C         .ADDRESS P.AAO                                  ;
                                                    00  00  47  4E  49  52  54  53  00070 P.AAQ:  .ASCII  \STRING\<0><0>             ;
                                                        010E0006  00078 P.AAP:  .LONG   17694726                                ;
                                                        00000000' 0007C         .ADDRESS P.AAQ                                  ;
                                                    00  00  45  43  52  55  4F  53  00080 P.AAS:  .ASCII  \SOURCE\<0><0>             ;
                                                        010E0006  00088 P.AAR:  .LONG   17694726                                ;
                                                        00000000' 0008C         .ADDRESS P.AAS                                  ;
                                    00  00  53  43  49  54  53  49  54  41  54  53  00090 P.AAU:  .ASCII  \STATISTICS\<0><0>         ;
                                                        010E000A  0009C P.AAT:  .LONG   17694730                                ;
                                                        00000000' 000A0         .ADDRESS P.AAU                                  ;
```

```
                                                00 00 00 54 43 41 58 45  000A4 P.AAW:   .ASCII  \EXACT\<0><0><0>
                                                   010E0005              000AC P.AAV:   .LONG   17694725
                                                   00000000'             000B0          .ADDRESS P.AAW
                                                            00 47 4F 4C  000B4 P.AAY:   .ASCII  \LOG\<0>
                                                   010E0003              000B8 P.AAX:   .LONG   17694723
                                                   00000000'             000BC          .ADDRESS P.AAY
                                                00 53 52 45 42 4D 55 4E  000C0 P.ABA:   .ASCII  \NUMBERS\<0>
                                                   010E0007              000C8 P.AAZ:   .LONG   17694727
                                                   00000000'             000CC          .ADDRESS P.ABA
                                                00 47 4E 49 44 41 45 48  000D0 P.ABC:   .ASCII  \HEADING\<0>
                                                   010E0007              000D8 P.ABB:   .LONG   17694727
                                                   00000000'             000DC          .ADDRESS P.ABC
                                    00 00 0C 47 4E 49 4E 49 41 4D 45 52  000E0 P.ABE:   .ASCII  \REMAINING\<0><0><0>
                                                   010E0009              000EC P.ABD:   .LONG   17694729
                                                   00000000'             000F0          .ADDRESS P.ABE
                                                00 00 54 55 50 54 55 4F  000F4 P.ABG:   .ASCII  \OUTPUT\<0><0>
                                                   010E0006              000FC P.ABF:   .LONG   17694726
                                                   00000000'             00100          .ADDRESS P.ABG
                                                00 00 00 48 43 54 41 4D  00104 P.ABI:   .ASCII  \MATCH\<0><0><0>
                                                   010E0005              0010C P.ABH:   .LONG   17694725
                                                   00000000'             00110          .ADDRESS P.ABI
                                                00 00 54 41 4D 52 4F 46  00114 P.ABK:   .ASCII  \FORMAT\<0><0>
                                                   010E0006              0011C P.ABJ:   .LONG   17694726
                                                   00000000'             00120          .ADDRESS P.ABK
                                                00 45 44 55 4C 43 58 45  00124 P.ABM:   .ASCII  \EXCLUDE\<0>
                                                   010E0007              0012C P.ABL:   .LONG   17694727
                                                   00000000'             00130          .ADDRESS P.ABM
3A 64 65 68 63 72 61 65 73 20 73 65 6C 69 46      00134 P.ABO:   .ASCII  \Files searched:        !10UL        Buffe\
20 20 4C 55 30 31 21 20 20 20 20 20 20 20 20      00143
                  65 66 66 75 42 20 20 20 20 20 20 00152
21 3A 74 6E 75 6F 63 20 4F 2F 49 20 64 65 72      0015C          .ASCII  \red I/O count:!10UL\<0>
                              00 4C 55 30 31 00   0016B
                                                   010E003B              00170 P.ABN:   .LONG   17694779
                                                   00000000'             00174          .ADDRESS P.ABO
65 68 63 72 61 65 73 20 73 64 72 6F 63 65 52      00178 P.ABQ:   .ASCII  \Records searched:      !10UL        Direc\
20 20 4C 55 30 31 21 20 20 20 20 20 3A 64 20      00187
                  63 65 72 69 44 20 20 20 20 20 20 00196
21 20 20 3A 74 6E 75 6F 63 20 4F 2F 49 20 74      001A0          .ASCII  \t I/O count:  !10UL\<0>
                              00 4C 55 30 31 00   001AF
                                                   010E003B              001B4 P.ABP:   .LONG   17694779
                                                   00000000'             001B8          .ADDRESS P.ABQ
72 61 65 73 20 73 72 65 74 63 61 72 61 68 43      001BC P.ABS:   .ASCII  \Characters searched:   !10UL        Page \
20 20 4C 55 30 31 21 20 20 20 3A 64 65 68 63      001CB
                  20 65 67 61 50 20 20 20 20 20 20 001DA
21 20 20 20 20 20 20 20 3A 73 74 6C 75 61 66      001E4          .ASCII  \faults:       !10UL\<0>
                              00 4C 55 30 31 00   001F3
                                                   010E003B              001F8 P.ABR:   .LONG   17694779
                                                   00000000'             001FC          .ADDRESS P.ABS
64 65 68 63 74 61 6D 20 73 64 72 6F 63 65 52      00200 P.ABU:   .ASCII  \Records matched:       !10UL        Elaps\
20 20 4C 55 30 31 21 20 20 20 20 20 20 20 3A      0020F
                  73 70 61 6C 45 20 20 20 20 20 20 0021E
55 33 21 3A 65 6D 69 74 20 55 50 43 20 64 65      00228          .ASCII  \ed CPU time:!3UL !2ZL:!2ZL:!2ZL.!2ZL\
5A 32 21 3A 4C 5A 32 21 3A 4C 5A 32 21 20 4C      00237
                              4C 5A 32 21 2E 4C   00246
                                                   010E004C              0024C P.ABT:   .LONG   17694796
                                                   00000000'             00250          .ADDRESS P.ABU
20 3A 64 65 74 6E 69 72 70 20 73 65 6E 69 4C      00254 P.ABW:   .ASCII  \Lines printed:         !10UL        Elaps\
```

```
20  20  4C  55  30  31  21  20  20  20  20  20  20  20  20  00263
                    73  70  61  6C  45  20  20  20  20  20  00272
00  53  41  21  20  20  20  3A  65  6D  69  74  20  64  65  0027C
                                                    00  0028B
                                            010E0036  0028C P.ABV:  .LONG   17694774
                                            00000000'  00290         .ADDRESS P.ABW

                                                        .ASCII  \ed time:    !AS\<0><0>

                                                    STARS_30=           P.AAK
                                                    NULL_STR=           P.AAL
                                                    WINDOW_STR=         P.AAN
                                                    STRING_STR=         P.AAP
                                                    SOURCE_STR=         P.AAR
                                                        .EXTRN  LIB$FILE_SCAN, LIB$LOOKUP_KEY
                                                        .EXTRN  LIB$PUT_OUTPUT, CLI$GET_VALUE
                                                        .EXTRN  CLI$PRESENT, OTS$CVT_TI_L
                                                        .EXTRN  STR$GET1_DX, STR$COPY_DX
                                                        .EXTRN  LIB$GET_VM, SRH$_BADEXCL
                                                        .EXTRN  SRH$_BADEXCLNAM
                                                        .EXTRN  SRH$_BADFORM, SRH$_BADMATCH
                                                        .EXTRN  SRH$_MATCHED, SRH$_NOMATCH
                                                        .EXTRN  SRH$_NOFILE, SRH$_NOMATCHES
                                                        .EXTRN  SRH$_NOSTRING, SRH$_NULLFILE
                                                        .EXTRN  SRH$_RFAERR, SRH$_TRUNCATE
                                                        .EXTRN  SRH$_WDW_MAXPREV
                                                        .EXTRN  SRH$_WDW_MAXPRM
                                                        .EXTRN  SYS$GETJPI, SYS$GETTIM
                                                        .EXTRN  SYS$CREATE, SYS$CONNECT
                                                        .EXTRN  SYS$TRNLOG, SYS$FAO
                                                        .EXTRN  SYS$ASCTIM, SYS$CLOSE

                                                        .PSECT  $CODE$,NOWRT,2

                                            0FFC 00000 MAIN:    .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11          ; 0434
                                    5B 00000000G   00   9E 00002        MOVAB   CLI$GET_VALUE, R11                    ;
                                    5A     0000'   CF   9E 00009        MOVAB   WINDOW_STR, R10                       ;
                                    59     0000'   CF   9E 0000E        MOVAB   TMP_DESC, R9                          ;
                                    58     0000'   CF   9E 00013        MOVAB   WDW_PRE_RECS, R8                      ;
                                    5E     C6D4    CE   9E 00018        MOVAB   -14636(SP), SP                        ;
                                             34    AA   9F 0001D        PUSHAB  P.AAT                                 ; 0486
                                00000000G    00        01   FB 00020        CALLS   #1, CLI$PRESENT                   ;
                                    0D08     C8        50   D0 00027        MOVL    R0, QUA_STATISTICS                ;
                                             3B        50   E9 0002C        BLBC    R0, 3$                            ;
                                                       7E   7C 0002F        CLRQ    -(SP)                             ; 0490
                                                       7E   D4 00031        CLRL    -(SP)                             ;
                                                 0238  C9   9F 00033        PUSHAB  STAT_JPI_BLOCK                    ;
                                                       7E   7C 00037        CLRQ    -(SP)                             ;
                                                       7E   D4 00039        CLRL    -(SP)                             ;
                                00000000G    00        07   FB 0003B        CALLS   #7, SYS$GETJPI                    ;
                                             56        50   D0 00042        MOVL    R0, STATUS                        ;
                                             0E        56   E9 00045        BLBC    STATUS, 1$                        ;
                                                 0D40  C8   9F 00048        PUSHAB  STAT_STARTTIME                    ; 0493
                                00000000G    00        01   FB 0004C        CALLS   #1, SYS$GETTIM                    ;
                                             56        50   D0 00053        MOVL    R0, STATUS                        ;
                                              3        56   E8 00056 1$:     BLBS    STATUS, 2$                        ;
                                                  05FD 31   00059        BRW     55$                               ;
                                    0D30      .8   0D20  C8   7D 0005C 2$:     MOVQ    STAT_JPI_BUFIO, STAT_BUFIO    ; 0497
                                    0D38     C8    0D28  C8   7D 00063        MOVQ    STAT_JPI_DIRIO, STAT_DIRIO    ; 0499
```

```
                00DC  C9      0124  CE  9E 0006A 3$:     MOVAB   INP_BUFFER, RAB+36                ; 0507
                  44  A8      0924  CE  9E 00071        MOVAB   OUT_BUFFER, OUTPUT_BUFF           ; 0508
                      44      AA  9F 00077              PUSHAB  P.AAV                             ; 0513
00000000G       00              01  FB 0007A            CALLS   #1, CLI$PRESENT
                  36  A8          50  90 00081        MOVB    R0, QUA_EXACT
                      50      AA  9F 00085              PUSHAB  P.AAX                             ; 0514
00000000G       00              01  FB 00088            CALLS   #1, CLI$PRESENT
                  37  A8          50  90 0008F        MOVB    R0, QUA_LOG
                      60      AA  9F 00093              PUSHAB  P.AAZ                             ; 0515
00000000G       00              01  FB 00096            CALLS   #1, CLI$PRESENT
                  38  A8          50  90 0009D        MOVB    R0, QUA_NUMBERS
                      70      AA  9F 000A1              PUSHAB  P.ABB                             ; 0516
00000000G       00              01  FB 000A4            CALLS   #1, CLI$PRESENT
                  39  A8          50  90 000AB        MOVB    R0, QUA_HEADING
                      F8      A9  9F 000AF              PUSHAB  MOD_DESC                          ; 0521
                              5A  DD 000B2              PUSHL   R10
                      6B          02  FB 000B4        CALLS   #2, CLI$GET_VALUE
                      53          50  D0 000B7        MOVL    R0, WDW_1
                      15          53  E9 000BA        BLBC    WDW_1, 4$
                          04  AE  9F 000BD              PUSHAB  WINDOW_1                          ; 0524
                          F8  A9  9F 000C0              PUSHAB  MOD_DESC
00000000G               00          02  FB 000C3        CALLS   #2, OTS$CVT_TI_L
                          28          50  E9 000CA        BLBC    R0, 5$
                          04  AE  D5 000CD              TSTL    WINDOW_1                          ; 0530
                              23  19 000D0              BLSS    5$
                          F8  A9  9F 000D2 4$:          PUSHAB  MOD_DESC                          ; 0540
                              5A  DD 000D5              PUSHL   R10
                      6B          02  FB 000D7        CALLS   #2, CLI$GET_VALUE
                      52          50  D0 000DA        MOVL    R0, WDW_2
                      2F          52  E9 000DD        BLBC    WDW_2, 6$
                          08  AE  9F 000E0              PUSHAB  WINDOW_2                          ; 0543
                          F8  A9  9F 000E3              PUSHAB  MOD_DESC
00000000G               00          02  FB 000E6        CALLS   #2, OTS$CVT_TI_L
                          05          50  E9 000ED        BLBC    R0, 5$
                              08  AE  D5 000F0          TSTL    WINDOW_2                          ; 0549
                              1A  18 000F3              BGEQ    6$
                          F8  A9  9F 000F5 5$:          PUSHAB  MOD_DESC                          ; 0552
                              01  DD 000F8              PUSHL   #1
                      00D71112    8F  DD 000FA          PUSHL   #14094610
00000000G               00          03  FB 00100        CALLS   #3, LIB$SIGNAL
                          50  0D71112    8F  D0 00107    MOVL    #282530066, R0                    ; 0553
                                      04  00010E       RET
                          F8  A9  9F 0010F 6$:          PUSHAB  MOD_DESC                          ; 0557
                              5A  DD 00112              PUSHL   R10
                      6B          02  FB 00114        CALLS   #2, CLI$GET_VALUE
                      08          50  E9 00117        BLBC    R0, 7$
                          50  00000000G  8F  D0 0011A    MOVL    #SRH$_WDW_MAXPRM, R0              ; 0559
                                      04  00121       RET
                      22          53  E9 00122 7$:     BLBC    WDW_1, 10$                        ; 0564
                      06          52  E9 00125        BLBC    WDW_2, 8$                         ; 0567
                      68      04  AE  7D 00128          MOVQ    WINDOW_1, WDW_PRE_RECS            ; 0570
                              19  11 0012C              BRB     10$                               ; 0567
                          04  AE  D5 0012E 8$:          TSTL    WINDOW_1                          ; 0575
                              06  12 00131              BNEQ    9$
                  33  A8          01  90 00133        MOVB    #1, WDW_ZERO                      ; 0577
                              0E  11 00137              BRB     10$
50              04  AE          01  C3 00139 9$:      SUBL3   #1, WINDOW_1, TMP                 ; 0582
```

```
                68              50              02  C7  0013E         DIVL3   #2, TMP, WDW_PRE_RECS                       ; 0583
          04    A8              50              68  C3  00142         SUBL3   WDW_PRE_RECS, TMP, WDW_SUB_RECS             ; 0584
                                                  68  D5  00147 10$:  TSTL    WDW_PRE_RECS                                ; 0589
                                                  05  14  00149         BGTR    11$
                                          04      A8  D5  0014B         TSTL    WDW_SUB_RECS                                ; 0590
                                                  04  15  0014E         BLEQ    12$
                                  34      A8      01  90  00150 11$:  MOVB    #1, WDW_FRAME                               ; 0592
                                        0084      CA  9F  00154 12$:  PUSHAB  P.ABD                                       ; 0594
                          00000000G     00          01  FB  00158         CALLS   #1, CLI$PRESENT
                                        04          50  E9  0015F         BLBC    R0, 13$
                                  04      A8      01  CE  00162         MNEGL   #1, WDW_SUB_RECS                            ; 0596
                                        F8      A9  9F  00166 13$:  PUSHAB  MOD_DESC                                    ; 0601
                                      0094      CA  9F  00169         PUSHAB  P.ABF
                                        6B          02  FB  0016D         CALLS   #2, CLI$GET_VALUE
                                  30      A8        50  90  00170         MOVB    R0, OUT_FILE_OPEN
                                        5B          50  E9  00174         BLBC    R0, 17$
                                0190      C9    F8    A9  90  00177         MOVB    MOD_DESC, OUTFAB+52                 ; 0604
                                0188      C9    FC    A9  D0  0017D         MOVL    MOD_DESC+4, OUTFAB+44               ; 0605
                                      015C      C9  9F  00183         PUSHAB  OUTFAB                                      ; 0606
                          00000000G     00          01  FB  00187         CALLS   #1, SYS$CREATE
                                        56          50  D0  0018E         MOVL    R0, STATUS
                                        06          56  E8  00191         BLBS    STATUS, 14$
                                      0168      C9    DD  00194         PUSHL   OUTFAB+12                           ; 0608
                                                  15  11  00198         BRB     15$
                                      01AC      C9  9F  0019A 14$:  PUSHAB  OUTRAB                                      ; 0609
                          00000000G     00          01  FB  0019E         CALLS   #1, SYS$CONNECT
                                        56          50  D0  001A5         MOVL    R0, STATUS
                                        16          56  E8  001A8         BLBS    STATUS, 16$
                                    01B8      C9    DD  001AB         PUSHL   OUTRAB+12                           ; 0611
                                      015C      C9  9F  001AF 15$:  PUSHAB  OUTFAB
                                                  56  DD  001B3         PUSHL   STATUS
                                00D710A2      8F      DD  001B5         PUSHL   #14094498
                    0000V       CF                    04  FB  001BB         CALLS   #4, FILE_ERROR
                                                      04  001C0         RET
                                  20      A8    0192    C9  3C  001C1 16$:  MOVZWL  OUTFAB+54, MAX_REC                  ; 0612
                                                20      A8  D5  001C7         TSTL    MAX_REC                             ; 0613
                                                  06  12  001CA         BNEQ    17$
                                  20      A8    FFFF    8F  3C  001CC         MOVZWL  #65535, MAX_REC
          31    A8              30      A8      33      A8  8B  001D2 17$:  BICB3   WDW_ZERO, OUT_FILE_OPEN, OUT_PRINTING  ; 0616
                                                F0      A9  9F  001D9         PUSHAB  STRING_DESC                         ; 0621
                                              00A4      CA  9F  001DC         PUSHAB  P.ABH
                                        6B                02  FB  001E0         CALLS   #2, CLI$GET_VALUE
                                        53        F4      A9  D0  001E3         MOVL    STRING_DESC+4, R3                   ; 0622
                                        52        F4      A9  D0  001E7         MOVL    STRING_DESC+4, R2
                                        51        F0      A9  3C  001EB         MOVZWL  STRING_DESC, R1
                                                      0000V   30  001EF         BSBW    UPCASE
                                                  0C      AE  9F  001F2         PUSHAB  KEYVAL                              ; 0624
                                                0214      C9  9F  001F5         PUSHAB  MATCH_KEYTBL
                                                  F0      A9  9F  001F9         PUSHAB  STRING_DESC
                          00000000G     00                03  FB  001FC         CALLS   #3, LIB$LOOKUP_KEY
                                        56                50  D0  00203         MOVL    R0, STATUS
                                        16                56  E8  00206         BLBS    STATUS, 18$
                                                    F0    A9  9F  00209         PUSHAB  STRING_DESC                         ; 0626
                                                          01  DD  0020C         PUSHL   #1
                                                          56  DD  0020E         PUSHL   STATUS
                                                          7E  D4  00210         CLRL    -(SP)
                                              00000000G   8F  DD  00212         PUSHL   #SRH$_BADMATCH
```

```
                00000000G  00          05  FB 00218          CALLS   #5, LIB$STOP
                       03  00      0C  AE  CF 0021F 18$:     CASEL   KEYVAL, #0, #3                          ;  0628
          0010       000A        0014      00224 19$:     .WORD   23$-19$,-
                                                                   22$-19$,-
                                                                   20$-19$,-
                                                                   21$-19$
                                          0E  11 0022C          BRB     23$
                          3E  A8            01  90 0022E 20$:     MOVB    #1, MAT_AND                         ;  0632
                                            08  11 00232          BRB     23$
                          3E  A8            01  90 00234 21$:     MOVB    #1, MAT_AND                         ;  0634
                          3F  A8            01  90 00238 22$:     MOVB    #1, MAT_NEGATE                      ;  0635
                              50            68  D0 0023C 23$:     MOVL    WDW_PRE_RECS, R0                    ;  0643
                                            07  12 0023F          BNEQ    24$
                          6C  A9      40    8F  88 00241          BISB2   #64, FAB+4                          ;  0645
                                            30  11 00246          BRB     25$
                          32  A8            01  90 00248 24$:     MOVB    #1, WDW_PREVIOUS                    ;  0648
                                      40    A8  9F 0024C          PUSHAB  RFA_VEC                             ;  0652
                              50            06  C4 0024F          MULL2   #6, R0
                          04  AE      06    A0  9E 00252          MOVAB   6(R0), 4(SP)
                                      04    AE  9F 00257          PUSHAB  4(SP)
                00000000G  00             02  FB 0025A          CALLS   #2, LIB$GET_VM
                              56            50  D0 00261          MOVL    R0, STATUS
                              11            56  E8 00264          BLBS    STATUS, 25$
                                            56  DD 00267          PUSHL   STATUS                              ;  0654
                                            7E  D4 00269          CLRL    -(SP)
                                 00000000G  8F  DD 0026B          PUSHL   #SRH$_WDW_MAXPREV
                00000000G  00             03  FB 00271          CALLS   #3, LIB$STOP
                                      F0    A9  9F 00278 25$:     PUSHAB  STRING_DESC                         ;  0660
                                    00B4    CA  9F 0027B          PUSHAB  P.ABJ
                              6B            02  FB 0027F          CALLS   #2, CLI$GET_VALUE
                              53        F4  A9  D0 00282          MOVL    STRING_DESC+4, R3                   ;  0661
                              52        F4  A9  D0 00286          MOVL    STRING_DESC+4, R2
                              51        F0  A9  3C 0028A          MOVZWL  STRING_DESC, R1
                                    0000V     30 0028E          BSBW    UPCASE
                                      24  A8  9F 00291          PUSHAB  FORMAT                              ;  0663
                                    01F0  C9  9F 00294          PUSHAB  FORMAT_KEYTBL
                                      F0  A9  9F 00298          PUSHAB  STRING_DESC
   00000000G  00                          03  FB 0029B          CALLS   #3, LIB$LOOKUP_KEY
                              56            50  D0 002A2          MOVL    R0, STATUS
                              16            56  E8 002A5          BLBS    STATUS, 26$
                                      F0  A9  9F 002A8          PUSHAB  STRING_DESC                         ;  0665
                                            01  DD 002AB          PUSHL   #1
                                            56  DD 002AD          PUSHL   STATUS
                                            7E  D4 002AF          CLRL    -(SP)
                                 00000000G  8F  DD 002B1          PUSHL   #SRH$_BADFORM
   00000000G  00                          05  FB 002B7          CALLS   #5, LIB$STOP
                              50        14  A8  D0 002BE 26$:     MOVL    CNT_INFILE, R0                      ;  0673
   00000088   8F                          50  D1 002C2          CMPL    R0, #136
                                          20  18 002C9          BGEQ    27$
                                    48 A840   7F 002CB          PUSHAQ  INFILE_DESC[R0]                     ;  0676
                          9E 020E0000       8F  D0 002CF          MOVL    #34471936, @(SP)+
                                    48 A840   7F 002D6          PUSHAQ  INFILE_DESC[R0]                     ;  0678
                                      20  AA  9F 002DA          PUSHAB  SOURCE_STR                          ;  0677
                              6B            02  FB 002DD          CALLS   #2, CLI$GET_VALUE                   ;  0678
                              56            50  D0 002E0          MOVL    R0, STATUS
                              05            56  E9 002E3          BLBC    STATUS, 27$
                                      14  A8  D6 002E6          INCL    CNT_INFILE                          ;  0681
```

```
                                 D3  11 002E9          BRB     26$                                     ; 0673
                              14 A8  D5 002EB 27$:     TSTL    CNT_INFILE                              ; 0687
                                 08  14 002EE          BGTR    28$                                     ;
                        50 00000000G 8F  D0 002F0          MOVL    #SRH$_NOFILE, R0                        ; 0689
                                 04  002F7              RET                                             ;
00000088                8F    18 A8  D1 002F8 28$:     CMPL    CNT_SRHSTR, #136                        ; 0697
                                 5E  18 00300          BGEQ    31$                                     ;
                              F0 A9  9F 00302          PUSHAB  STRING_DESC                             ; 0700
                              10 AA  9F 00305          PUSHAB  STRING_STR                              ;
                        6B          02  FB 00308          CALLS   #2, CLI$GET_VALUE                       ;
                        56          50  D0 0030B          MOVL    R0, STATUS                              ;
                        4F          56  E9 0030E          BLBC    STATUS, 31$                             ;
                        50    18 A8  D0 00311          MOVL    CNT_SRHSTR, R0                          ; 0706
                        50  0488 C840  7E 00315          MOVAQ   SRHSTR_DESC[R0], R0                     ;
                        60 020E0000 8F  D0 0031B          MOVL    #34471936, (R0)                         ;
                        0E    36 A8  E9 00322          BLBC    QUA_EXACT, 29$                          ; 0708
                              F0 A9  9F 00326          PUSHAB  STRING_DESC                             ; 0710
                                 50  DD 00329          PUSHL   R0                                      ;
00000000G               00          02  FB 0032B          CALLS   #2, STR$COPY_DX                         ;
                                 27  11 00332          BRB     30$                                     ;
                                 50  DD 00334 29$:     PUSHL   R0                                      ; 0714
                              F0 A9  9F 00336          PUSHAB  STRING_DESC                             ; 0713
00000000G               00          02  FB 00339          CALLS   #2, STR$GET1_DX                         ; 0714
                        50    18 A8  D0 00340          MOVL    CNT_SRHSTR, R0                          ; 0718
                           048C C840  7F 00344          PUSHAQ  SRHSTR_DESC+4[R0]                       ; 0716
                        53          9E  D0 00349          MOVL    @(SP)+, R3                              ;
                        52    F4 A9  D0 0034C          MOVL    STRING_DESC+4, R2                       ;
                           0488 C840  7F 00350          PUSHAQ  SRHSTR_DESC[R0]                         ;
                        51          9E  3C 00355          MOVZWL  @(SP)+, R1                              ;
                                 0000V 30 00358          BSBW    UPCASE                                  ;
                              18 A8  D6 0035B 30$:     INCL    CNT_SRHSTR                              ; 0721
                                 98  11 0035E          BRB     28$                                     ; 0697
                              18 A8  D5 00360 31$:     TSTL    CNT_SRHSTR                              ; 0727
                                 08  14 00363          BGTR    32$                                     ;
                        50 00000000G 8F  D0 00365          MOVL    #SRH$_NOSTRING, R0                      ; 0729
                                 04  0036C              RET                                             ;
                        57    18 AE  9E 0036D 32$:     MOVAB   RESULT_DESC+4, R7                       ; 0752
                  00000088    8F  1C A8  D1 00371 33$:     CMPL    CNT_EXCLUDE, #136                       ; 0737
                                 03  19 00379          BLSS    35$                                     ;
                                 00B6  31 0037B 34$:     BRW     43$                                     ;
                              F0 A9  9F 0037E 35$:     PUSHAB  STRING_DESC                             ; 0745
                           00C4 CA  9F 00381          PUSHAB  P.ABL                                   ;
                        6B          02  FB 00385          CALLS   #2, CLI$GET_VALUE                       ;
                        56          50  D0 00388          MOVL    R0, STATUS                              ;
                        ED          56  E9 0038B          BLBC    STATUS, 34$                             ;
1C  AE          F0  A9              08  28 0038E          MOVC3   #8, STRING_DESC, INTER_DESC             ; 0751
                14  AE              69  D0 00394          MOVL    TMP_DESC, RESULT_DESC                   ; 0752
                14  AE        FF    8F  9B 00398 36$:     MOVZBW  #255, RESULT_DESC                       ; 0755
                18  AE        24    AE  9E 0039D          MOVAB   BUF, RESULT_DESC+4                      ; 0756
                                    7E  7C 003A2          CLRQ    -(SP)                                   ; 0757
                                    7E  D4 003A4          CLRL    -(SP)                                   ;
                              20    AE  9F 003A6          PUSHAB  RESULT_DESC                             ;
                              24    AE  9F 003A9          PUSHAB  RESULT_DESC                             ;
                              30    AE  9F 003AC          PUSHAB  INTER_DESC                              ;
                  00000000G   00          06  FB 003AF          CALLS   #6, SYS$TRNLOG                          ;
                        56          50  D0 003B6          MOVL    R0, STATUS                              ;
                        09          56  E8 003B9          BLBS    STATUS, 37$                             ; 0758
```

```
                                    56  DD 003BC          PUSHL   STATUS                                                  ; 0760
                00000000G     00    01  FB 003BE          CALLS   #1, LIB$STOP                                          ;
          1C    AE      14      AE  08  28 003C5 37$:     MOVC3   #8, RESULT_DESC, INTER_DESC                           ; 0761
                00000629      8F  56  D1 003CB          CMPL    STATUS, #1577                                         ; 0763
                                    C4  12 003D2          BNEQ    36$                                                   ;
          18    BE      14      AE  3A  3A 003D4          LOCC    #58, RESULT_DESC, @RESULT_DESC+4                      ; 0769
                                    02  12 003DA          BNEQ    38$                                                   ;
                                    51  D4 003DC          CLRL    R1                                                    ;
                                    51  D5 003DE 38$:     TSTL    R1                                                    ; 0770
                                    1D  12 003E0          BNEQ    41$                                                   ;
          18    BE      14      AE      5D  8F  3A 003E2  LOCC    #93, RESULT_DESC, @RESULT_DESC+4                      ; 0771
                                    02  12 003E9          BNEQ    39$                                                   ;
                                    51  D4 003EB          CLRL    R1                                                    ;
                                    51  D5 003ED 39$:     TSTL    R1                                                    ; 0772
                                    0E  12 003EF          BNEQ    41$                                                   ;
          18    BE      14      AE  3E  3A 003F1          LOCC    #62, RESULT_DESC, @RESULT_DESC+4                      ; 0773
                                    02  12 003F7          BNEQ    40$                                                   ;
                                    51  D4 003F9          CLRL    R1                                                    ;
                                    51  D5 003FB 40$:     TSTL    R1                                                    ; 0774
                                    12  13 003FD          BEQL    42$                                                   ;
                                14      AE  9F 003FF 41$:     PUSHAB  RESULT_DESC                                       ; 0776
                                    01  DD 00402          PUSHL   #1                                                    ;
                            00000000G   8F  DD 00404          PUSHL   #SRH$_BADEXCL                                     ;
                00000000G     00    03  FB 0040A          CALLS   #3, LIB$STOP                                          ;
                              50      1C  A8  D0 00411 42$:     MOVL    CNT_EXCLUDE, R0                               ; 0781
                                  08C8 C840 7F 00415          PUSHAQ  EXCLUDE_DESC[R0]                                  ;
                              9E 020E0000   8F  D0 0041A      MOVL    #34471936, @(SP)+                                 ;
                                  08C8 C840 7F 00421          PUSHAQ  EXCLUDE_DESC[R0]                                  ; 0782
                                      18    AE  9F 00426      PUSHAB  RESULT_DESC                                       ;
                          0000V     CF      02  FB 00429      CALLS   #2, VALIDATE_EXCLUDE_FILE                         ;
                                      1C    A8  D6 0042E      INCL    CNT_EXCLUDE                                       ; 0783
                                         FF3D 31 00431        BRW     33$                                               ; 0737
                                      3B    A8  B4 00434 43$:     CLRW    FOUND_FILE                                    ; 0788
                                      10    AE  D4 00437      CLRL    SCAN_CONTEXT                                      ; 0790
                              53      14    A8  D0 0043A      MOVL    CNT_INFILE, R3                                    ; 0791
                              52              01  CE 0043E      MNEGL   #1, IFX                                           ;
                                              2B  11 00441      BRB     45$                                               ;
                                      4C A842 7F 00443 44$:     PUSHAQ  INFILE_DESC+4[IFX]                                ; 0794
                          0094    C9          9E  D0 00447      MOVL    @(SP)+, FAB+44                                    ;
                                      48 A842 7F 0044C          PUSHAQ  INFILE_DESC[IFX]                                  ; 0795
                          009C    C9          9E  90 00450      MOVB    @(SP)+, FAB+52                                    ;
                                      10    AE  9F 00455      PUSHAB  SCAN_CONTEXT                                      ; 0796
                                  0000V     CF  9F 00458      PUSHAB  SCAN_FAILURE                                      ;
                                  0000V     CF  9F 0045C      PUSHAB  SCAN_SUCCESS                                      ;
                                      68    A9  9F 00460      PUSHAB  FAB                                               ;
                        00000000G     00        04  FB 00463  CALLS   #4, LIB$FILE_SCAN                                 ;
                                  0098    C9  D4 0046A      CLRL    FAB+48                                            ; 0800
                    D1                52      53  F2 0046E 45$:     AOBLSS  R3, IFX, 44$                              ; 0791
                                      03  0D08 C8  E8 00472      BLBS    QUA_STATISTICS, 46$                               ; 0807
                                           01AC 31 00477        BRW     52$                                               ;
                                              7E  7C 0047A 46$:     CLRQ    -(SP)                                         ; 0820
                                              7E  D4 0047C      CLRL    -(SP)                                             ;
                                  0238        C9  9F 0047E      PUSHAB  STAT_JPI_BLOCK                                    ;
                                              7E  7C 00482      CLRQ    -(SP)                                             ;
                                              7E  D4 00484      CLRL    -(SP)                                             ;
                        00000000G     00        07  FB 00486  CALLS   #7, SYS$GETJPI                                   ;
                                      52        50  D0 0048D  MOVL    R0, STAT                                          ;
```

```
                60              52  E9 00490          BLBC    STAT, 47$                                                   ;
            04  A9      00B4    CE  9E 00493          MOVAB   OUT_BUF, TMP_DESC+4                                         ; 0824
                FO      AA      9F 00499              PUSHAB  NULL_STR                                                    ; 0825
    00000000G   00              01  FB 0049C          CALLS   #1, LIB$PUT_OUTPUT                                          ;
                69          50  8F  9B 004A3          MOVZBW  #80, TMP_DESC                                               ; 0827
7E      0D20    C8      0D30    C8  C3 004A7          SUBL3   STAT_BUFIO, STAT_JPI_BUFIO, -(SP)                           ; 0829
                        0D14    C8  DD 004AF          PUSHL   STAT_TOTFIL                                                 ;
                                59  DD 004B3          PUSHL   R9                                                          ;
                                59  DD 004B5          PUSHL   R9                                                          ;
                        0108    CA  9F 004B7          PUSHAB  P.ABN                                                       ;
    00000000G   00              05  FB 004BB          CALLS   #5, SYS$FAO                                                 ;
                52              50  D0 004C2          MOVL    R0, STAT                                                    ;
                59              52  E9 004C5          BLBC    STAT, 48$                                                   ;
                                59  DD 004C8          PUSHL   R9                                                          ; 0832
    00000000G   00              01  FB 004CA          CALLS   #1, LIB$PUT_OUTPUT                                          ;
                69          50  8F  9B 004D1          MOVZBW  #80, TMP_DESC                                               ; 0834
7E      0D28    C8      0D38    C8  C3 004D5          SUBL3   STAT_DIRIO, STAT_JPI_DIRIO, -(SP)                           ; 0836
                        0D10    C8  DD 004DD          PUSHL   STAT_TOTREC                                                 ;
                                59  DD 004E1          PUSHL   R9                                                          ;
                                59  DD 004E3          PUSHL   R9                                                          ;
                        014C    CA  9F 004E5          PUSHAB  P.ABP                                                       ;
    00000000G   00              05  FB 004E9          CALLS   #5, SYS$FAO                                                 ;
                52              50  D0 004F0          MOVL    R0, STAT                                                    ;
                2B              52  E9 004F3  47$:     BLBC    STAT, 48$                                                   ;
                                59  DD 004F6          PUSHL   R9                                                          ; 0839
    00000000G   00              01  FB 004F8          CALLS   #1, LIB$PUT_OUTPUT                                          ;
                69          50  8F  9B 004FF          MOVZBW  #80, TMP_DESC                                               ; 0841
7E      0D2C    C8      0D3C    C8  C3 00503          SUBL3   STAT_PAGEFLTS, STAT_JPI_PAGEFLTS, -(SP)                     ; 0843
                        0D0C    C8  DD 0050B          PUSHL   STAT_TOTCHR                                                 ;
                                59  DD 0050F          PUSHL   R9                                                          ;
                                59  DD 00511          PUSHL   R9                                                          ;
                        0190    CA  9F 00513          PUSHAB  P.ABR                                                       ;
    00000000G   00              05  FB 00517          CALLS   #5, SYS$FAO                                                 ;
                52              50  D0 0051E          MOVL    R0, STAT                                                    ;
                77              52  E9 00521  48$:     BLBC    STAT, 49$                                                   ;
                                59  DD 00524          PUSHL   R9                                                          ; 0846
    00000000G   00              01  FB 00526          CALLS   #1, LIB$PUT_OUTPUT                                          ;
                69          50  8F  9B 0052D          MOVZBW  #80, TMP_DESC                                               ; 0848
    50  0D24    C8      0D34    C8  C3 00531          SUBL3   STAT_CPUTIM, STAT_JPI_CPUTIM, TIM                           ; 0849
7E  00          50              01  7A 00539          EMUL    #1, TIM, #0, -(SP)                                          ; 0856
6E  7E              8E  00000064    8F  7B 0053E      EDIV    #100, (SP)+, -(SP), (SP)
    51              50  00000064    8F  C7 00547      DIVL3   #100, TIM, R1
7E  00              51              01  7A 0054F      EMUL    #1, R1, #0, -(SP)
6E  7E              8E              3C  7B 00554      EDIV    #60, (SP)+, -(SP), (SP)
    51              50  00001770    8F  C7 00559      DIVL3   #6000, TIM, R1
7E  00              51              01  7A 00561      EMUL    #1, R1, #0, -(SP)
6E  7E              8E              3C  7B 00566      EDIV    #60, (SP)+, -(SP), (SP)
    51              50  00057E40    8F  C7 0056B      DIVL3   #360000, TIM, R1
7E  00              51              01  7A 00573      EMUL    #1, R1, #0, -(SP)
6E  7E              8E              18  7B 00578      EDIV    #24, (SP)+, -(SP), (SP)
    7E              50  0083D600    8F  C7 0057D      DIVL3   #8640000, TIM, -(SP)
                            0D18    C8  DD 00585      PUSHL   STAT_TOTMAT                                                 ;
                                59  DD 00589          PUSHL   R9                                                          ;
                                59  DD 0058B          PUSHL   R9                                                          ;
                        01E4    CA  9F 0058D          PUSHAB  P.ABT                                                       ;
    00000000G   00              09  FB 00591          CALLS   #9, SYS$FAO                                                 ;
                52              50  D0 00598          MOVL    R0, STAT                                                    ;
```

```
                                7B              52  E9 00059B 49$:     BLBC    STAT, 50$                                       ;
                                                59  DD 0059E          PUSHL   R9                                              ; 0859
                    00000000G   00              01  FB 005A0          CALLS   #1, LIB$PUT_OUTPUT
                          0104  CE              10  D0 005A7          MOVL    #16, TIM_DESC                                   ; 0860
                          0108  CE        010C  CE  9E 005AC          MOVAB   TIM_BUF, TIM_DESC+4                             ; 0861
                                        011C  CE  9F 005B3          PUSHAB  QUADTIME                                        ; 0862
                    00000000G   00              01  FB 005B7          CALLS   #1, SYS$GETTIM
                                52              50  D0 005BE          MOVL    R0, STAT                                        ;
                                55              52  E9 005C1          BLBC    STAT, 50$                                       ;
011C    CE          0D40        C8        011C  CE  C3 005C4          SUBL3   QUADTIME, STAT_STARTTIME, QUADTIME              ; 0865
                                50        0D44  C8  D0 005CE          MOVL    STAT_STARTTIME+4, R0                            ;
                                50        0120  CE  D9 005D3          SBWC    QUADTIME, R0                                    ;
                          0120  CE              50  D0 005D8          MOVL    R0, QUADTIME                                    ;
                                              7E  D4 005DD          CLRL    -(SP)                                           ; 0866
                                        0120  CE  9F 005DF          PUSHAB  QUADTIME                                        ;
                                        010C  CE  9F 005E3          PUSHAB  TIM_DESC                                        ;
                                        0110  CE  9F 005E7          PUSHAB  TIM_DESC                                        ;
                    00000000G   00              04  FB 005EB          CALLS   #4, SYS$ASCTIM                                  ;
                                52              50  D0 005F2          MOVL    R0, STAT                                        ;
                                21              52  E9 005F5          BLBC    STAT, 50$                                       ;
                                69        50    8F  9B 005F8          MOVZBW  #80, TMP_DESC                                   ; 0870
                                        0104  CE  9F 005FC          PUSHAB  TIM_DESC                                        ; 0872
                                        0D1C  C8  DD 00600          PUSHL   STAT_TOTPUT                                     ;
                                              59  DD 00604          PUSHL   R9                                              ;
                                              59  DD 00606          PUSHL   R9                                              ;
                                        0224  CA  9F 00608          PUSHAB  P.ABV                                           ;
                    00000000G   00              05  FB 0060C          CALLS   #5, SYS$FAO                                     ;
                                52              50  D0 00613          MOVL    R0, STAT                                        ;
                                04              52  E8 00616          BLBS    STAT, 51$                                       ;
                                50              52  D0 00619 50$:     MOVL    STAT, R0                                        ; 0874
                                                04  0061C            RET
                                              59  DD 0061D 51$:     PUSHL   R9                                              ; 0875
                00000000G   00              01  FB 0061F          CALLS   #1, LIB$PUT_OUTPUT
                                0B        30    A8  E9 00626 52$:     BLBC    OUT_FILE_OPEN, 53$                              ; 0880
                                        015C  C9  9F 0062A          PUSHAB  OUTFAB                                          ; 0882
                00000000G   00              01  FB 0062E          CALLS   #1, SYS$CLOSE
                                56              01  D0 00635 53$:     MOVL    #1, STATUS                                      ; 0884
                                1D        3C    A8  E8 00638          BLBS    FOUND_ANY, 55$                                  ; 0885
                                12        3B    A8  E9 0063C          BLBC    FOUND_FILE, 54$                                 ; 0887
                                56  00000000G   8F  D0 00640          MOVL    #SRH$_NOMATCHES, STATUS                         ; 0889
                                              56  DD 00647          PUSHL   STATUS
                00000000G   00              01  FB 00649          CALLS   #1, LIB$SIGNAL
                                              07  11 00650          BRB     55$                                             ;
                                56  00000000G   8F  D0 00652 54$:     MOVL    #SRH$_NOFILE, STATUS                            ; 0891
                                50              56  D0 00659 55$:     MOVL    STATUS, R0                                      ; 0893
                                                04  0065C            RET                                                     ; 0894

; Routine Size:  1629 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  901          0895  1 ROUTINE scan_success : NOVALUE =
;  902          0896  2 BEGIN
;  903          0897  2
;  904          0898  2 ! Make sure that this file isn't being excluded
;  905          0899  2 !
;  906          0900  2 IF NOT exclude_file()                    ! Check exclude list
;  907          0901  2 THEN
;  908          0902  3     BEGIN
;  909          0903  3     found_file = true;
;  910          0904  3     search_one_file();                   ! Open & search a file
;  911          0905  2     END;
;  912          0906  2
;  913          0907  2 RETURN;
;  914          0908  1 END;


                                        0000 00000 SCAN_SUCCESS:
                                                   .WORD   Save nothing                    ; 0895
                0000V  CF                00  FB 00002       CALLS   #0, EXCLUDE_FILE        ; 0900
                       0A                50  E8 00007       BLBS    R0, 1$                  ;
                0000'  CF                01  90 0000A       MOVB    #1, FOUND_FILE          ; 0903
                0000V  CF                00  FB 0000F       CALLS   #0, SEARCH_ONE_FILE     ; 0904
                                            04 00014 1$:    RET                             ; 0908

; Routine Size:  21 bytes,    Routine Base:  $CODE$ + 065D
```

```
;  916          0909  1 ROUTINE scan_failure : NOVALUE =
;  917          0910  2 BEGIN
;  918          0911  2
;  919          0912  2     file_error (srh$_openin, .fab [fab$l_sts], fab, .fab [fab$l_stv]);
;  920          0913  2
;  921          0914  2 RETURN;
;  922          0915  1 END;


                                        0000 00000 SCAN_FAILURE:
                                                           .WORD   Save nothing                                    ; 0909
                                    0000'  CF  DD 00002    PUSHL   FAB+12                                          ; 0912
                                    0000'  CF  9F 00006    PUSHAB  FAB                                             ;
                                    0000'  CF  DD 0000A    PUSHL   FAB+8                                           ;
                                00D71098   8F  DD 0000E    PUSHL   #14094488                                       ;
                          0000V  CF            04  FB 00014    CALLS   #4, FILE_ERROR                             ;
                                               04 00019    RET                                                     ; 0915

; Routine Size:  26 bytes,    Routine Base:  $CODE$ + 0672
```

```
;  924     0916  1 ROUTINE validate_exclude_file ( in_desc  : REF BLOCK [8, BYTE],
;  925     0917  1                                  out_desc : REF BLOCK [8, BYTE]) : NOVALUE =
;  926     0918  1
;  927     0919  1 !++
;  928     0920  1 ! Functional description
;  929     0921  1 !
;  930     0922  1 !       This routine checks to see if string_desc is a valid filename
;  931     0923  1 !
;  932     0924  1 ! Calling sequence
;  933     0925  1 !
;  934     0926  1 !       validate_exclude_file (in_desc, out_desc)
;  935     0927  1 !
;  936     0928  1 ! Input parameters
;  937     0929  1 !
;  938     0930  1 !       in_desc     - Address of any class string descriptor containing a
;  939     0931  1 !                     filename to be validated
;  940     0932  1 !
;  941     0933  1 ! Implicit inputs
;  942     0934  1 !
;  943     0935  1 !       none
;  944     0936  1 !
;  945     0937  1 ! Output parameters
;  946     0938  1 !
;  947     0939  1 !       out_desc - Address of any class descriptor to receive validated name.
;  948     0940  1 !                  This copy will NOT have node, device or directory.  Leading
;  949     0941  1 !                  zeroes will be stripped from version numbers.  If a name
;  950     0942  1 !                  of the form NAME.TYP; or NAME.TYP;0 is given, the version
;  951     0943  1 !                  will also be stripped.
;  952     0944  1 !
;  953     0945  1 ! Implicit outputs
;  954     0946  1 !
;  955     0947  1 !       none
;  956     0948  1 !
;  957     0949  1 ! Routine value
;  958     0950  1 !
;  959     0951  1 !       TRUE    - The filename is valid
;  960     0952  1 !       error   - The filename does not parse without errors, the RMS error
;  961     0953  1 !                 code is returned
;  962     0954  1 !
;  963     0955  1 ! Side effects
;  964     0956  1 !
;  965     0957  1 !       Errors will be signalled
;  966     0958  1 !
;  967     0959  1 !--
;  968     0960  1
;  969     0961  2 BEGIN
;  970     0962  2
;  971     0963  2 OWN_DZRO
;  972     0964  2     val_exp     : VECTOR [nam$c_maxrss, BYTE],  ! Expanded name string
;  973     0965  2     val_res     : VECTOR [nam$c_maxrss, BYTE];  ! Resultant name string
;  974     0966  2
;  975     0967  2 OWN_CRF
;  976   P 0968  2     val_nam     : $NAM (                        ! File name block
;  977   P 0969  2                          RSA = val_res,         !  Result name addr
;  978   P 0970  2                          RSS = nam$c_maxrss,    !  Result name size
;  979   P 0971  2                          ESA = val_exp,         !  Expanded name addr
;  980     0972  2                          ESS = nam$c_maxrss),   !  Expanded name size
```

```
;  981        P 0973  2     val_fab     : $FAB(                                 ! Input file FAB
;  982        P 0974  2                          DNA = UPLIT BYTE('*.*'),        !  Default file spec.
;  983        P 0975  2                          DNS = 3,                        !   (DNM doesn't like *'s)
;  984          0976  2                          NAM = val_nam);                 !  Name block
;  985          0977  2
;  986          0978  2 LOCAL
;  987          0979  2     lve,                    ! Length from version to end
;  988          0980  2     len,                    ! The current length
;  989          0981  2     adr,                    ! The current start
;  990          0982  2     p,                      ! A pointer into the string
;  991          0983  2     status;
;  992          0984  2
;  993          0985  2 val_fab [fab$l_fna] = .in_desc [dsc$a_pointer]; ! Set name addr
;  994          0986  2 val_fab [fab$b_fns] = .in_desc [dsc$w_length];  ! Set name size
;  995          0987  2
;  996          0988  3 IF NOT (status = $PARSE(FAB = val_fab))         ! Parse the file spec
;  997          0989  2 THEN
;  998          0990  3     BEGIN
;  999          0991  3     file_error (srh$_badexclnam, .status, val_fab, .val_fab [fab$l_stv]);
; 1000          0992  3     RETURN;
; 1001          0993  2     END;
; 1002          0994  2
; 1003          0995  2 status = $SEARCH(FAB = val_fab);                ! Parse the file spec
; 1004          0996  3 IF (NOT .status ) AND (.status NEQ RMS$_FNF)
; 1005          0997  2 THEN
; 1006          0998  3     BEGIN
; 1007          0999  3     file_error (srh$_badexclnam, .status, val_fab, .val_fab [fab$l_stv]);
; 1008          1000  3     RETURN;
; 1009          1001  2     END;
; 1010          1002  2
; 1011          1003  2 len = .val_nam [nam$b_esl];             ! Get length of expanded string
; 1012          1004  2 adr = .val_nam [nam$l_esa];             ! Get address of expanded string
; 1013          1005  2 p = locate_filename (.len, .adr);       ! Skip device and directory
; 1014          1006  2 len = .len - (.p - .adr);               ! Adjust the length
; 1015          1007  2 adr = .p;                               ! And move the pointer
; 1016          1008  2 p = locate_version (.len, .adr);        ! Find the version number
; 1017          1009  2 lve = .len - (.p - .adr);               ! Compute remaining length
; 1018          1010  2 WHILE 1                         ! Trim leading zeroes in length
; 1019          1011  2 DO
; 1020          1012  3     BEGIN
; 1021          1013  4     IF (.lve GTR 2) AND (CH$RCHAR(.p+1) EQL '0')
; 1022          1014  3     THEN
; 1023          1015  4         BECIN
; 1024          1016  4         lve = .lve - 1;             ! Remove one from remaining length
; 1025          1017  4         len = .len - 1;             ! Also trim one from full length
; 1026          1018  4         CH$MOVE(.lve, .p+2, .p+1);      ! Slide it over one byte
; 1027          1019  4         END
; 1028          1020  3     ELSE
; 1029          1021  3         EXITLOOP;
; 1030          1022  2     END;
; 1031          1023  2
; 1032          1024  3 IF   (.lve LEQ 1)
; 1033          1025  3  OR ((.lve EQL 2) AND (CH$RCHAR(.p+1) EQL '0'))
; 1034          1026  2 THEN
; 1035          1027  2     len = .len - .lve;
; 1036          1028  2 tmp_desc [dsc$w_length]  = .len;
; 1037          1029  2 tmp_desc [dsc$a_pointer] = .adr;
```

```
;  1038          1030  2 str$copy_dx (.out_desc, tmp_desc);      ! Copy to the output descriptor
;  1039          1031  2
;  1040          1032  2 RETURN;
;  1041          1033  1 END;


                                        .PSECT  $OWN$_CRF,NOEXE,2

                                02  0027C VAL_NAM:.BYTE   2
                                60  0027D         .BYTE   96
                                FF  0027E         .BYTE   -1
                                00  0027F         .BYTE   0
                         00000000'  00280         .ADDRESS VAL_RES
                                00  00284         .BYTE   0
                                00  00285         .BYTE   0
                                FF  00286         .BYTE   -1
                                00  00287         .BYTE   0
                         00000000'  00288         .ADDRESS VAL_EXP
                         00000000   0028C         .LONG   0
                             0000#  00290         .WORD   0[8]
                             0000#  002A0         .WORD   0[3]
                             0000#  002A6         .WORD   0[3]
                         00000000   002AC         .LONG   0
                         00000000   002B0         .LONG   0
                                00  002B4         .BYTE   0
                                00  002B5         .BYTE   0
                                00  002B6         .BYTE   0
                                00  002B7         .BYTE   0
                                00  002B8         .BYTE   0
                                00  002B9         .BYTE   0
                                00#  002BA        .BYTE   0[2]
                         00000000   002BC         .LONG   0
                         00000000   002C0         .LONG   0
                         00000000   002C4         .LONG   0
                         00000000   002C8         .LONG   0
                         00000000   002CC         .LONG   0
                         00000000   002D0         .LONG   0
                         00000000#  002D4         .LONG   0[2]
                                03  002DC VAL_FAB:.BYTE   3
                                50  002DD         .BYTE   80
                              0000  002DE         .WORD   0
                         00000000   002E0         .LONG   0
                         00000000   002E4         .LONG   0
                         00000000   002E8         .LONG   0
                         00000000   002EC         .LONG   0
                              0000  002F0         .WORD   0
                                02  002F2         .BYTE   2
                                00  002F3         .BYTE   0
                         00000000   002F4         .LONG   0
                                00  002F8         .BYTE   0
                                00  002F9         .BYTE   0
                                00  002FA         .BYTE   0
                                02  002FB         .BYTE   2
                         00000000   002FC         .LONG   0
                         00000000   00300         .LONG   0
                         00000000'  00304         .ADDRESS VAL_NAM
```

```
                00000000  00308          .LONG   0
                00000000' 0030C          .ADDRESS P.ABX
                      00  00310          .BYTE   0
                      03  00311          .BYTE   3
                    0000  00312          .WORD   0
                00000000  00314          .LONG   0
                    0000  00318          .WORD   0
                      00  0031A          .BYTE   0
                      00  0031B          .BYTE   0
                00000000  0031C          .LONG   0
                00000000  00320          .LONG   0
                    0000  00324          .WORD   0
                      00  00326          .BYTE   0
                      00  00327          .BYTE   0
                00000000  00328          .LONG   0

                                         .PSECT  $OWN$_DZRO,NOEXE,2

                          01148 VAL_EXP: .BLKB   255
                          01247          .BLKB   1
                          01248 VAL_RES: .BLKB   255

                                         .PSECT  $PLIT$,NOWRT,NOEXE,2

                2A  2E  2A  00294 P.ABX:  .ASCII  \*.*\                                    ;

                                         .EXTRN  SYS$PARSE, SYS$SEARCH

                                         .PSECT  $CODE$,NOWRT,2

                          07FC 00000 VALIDATE_EXCLUDE_FILE:
                                         .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10          ; 0916
                      5A      0000'  CF  9E 00002          MOVAB   VAL_FAB, R10
                      50        04   AC  D0 00007          MOVL    IN_DESC, R0             ; 0985
                    2C  AA      04   A0  D0 0000B          MOVL    4(R0), VAL_FAB+44
                    34  AA           60  90 00010          MOVB    (R0), VAL_FAB+52        ; 0986
                                     5A  DD 00014          PUSHL   R10                     ; 0988
                00000000G   00       01  FB 00016          CALLS   #1, SYS$PARSE
                      52            50  D0 0001D          MOVL    R0, STATUS
                      18            52  E9 00020          BLBC    STATUS, 1$
                                     5A  DD 00023          PUSHL   R10                     ; 0995
                00000000G   00       01  FB 00025          CALLS   #1, SYS$SEARCH
                      52            50  D0 0002C          MOVL    R0, STATUS
                      1C            52  E8 0002F          BLBS    STATUS, 2$              ; 0996
                00018292    8F        52  D1 00032          CMPL    STATUS, #98962
                                      13  13 00039          BEQL    2$
                              0C      AA  DD 0003B 1$:      PUSHL   VAL_FAB+12              ; 0999
                            0404      8F  BB 0003E          PUSHR   #^M<R2,R10>
                      00000000G       8F  DD 00042          PUSHL   #SRH$_BADEXCLNAM
                    0000V   CF            04  FB 00048          CALLS   #4, FILE_ERROR
                                          04  0004D          RET                             ; 0998
                      58        AB    AA  9A 0004E 2$:      MOVZBL  VAL_NAM+11, LEN         ; 1003
                      56        AC    AA  D0 00052          MOVL    VAL_NAM+12, ADR         ; 1004
                                        56  DD 00056          PUSHL   ADR                     ; 1005
                                        58  DD 00058          PUSHL   LEN
                    0000V   CF            02  FB 0005A          CALLS   #2, LOCATE_FILENAME
                      57            50  D0 0005F          MOVL    R0, P
```

```
                50              56              57  C3 00062          SUBL3   P, ADR, R0                      ; 1006
                                58              50  C0 00066          ADDL2   R0, LEN                         ;
                                56              57  D0 00069          MOVL    P, ADR                          ; 1007
                                                56  DD 0006C          PUSHL   ADR                             ; 1008
                                                58  DD 0006E          PUSHL   LEN                             ;
                        0000V   CF              02  FB 00070          CALLS   #2, LOCATE_VERSION              ;
                                57              50  D0 00075          MOVL    R0, P                           ;
                50              56              57  C3 00078          SUBL3   P, ADR, R0                      ; 1009
                59              50              58  C1 0007C          ADDL3   LEN, R0, LVE                    ;
                                02              59  D1 00080 3$:      CMPL    LVE, #2                         ; 1013
                                                12  15 00083          BLEQ    4$                              ;
                                30      01      A7  91 00085          CMPB    1(P), #48                       ;
                                                0C  12 00089          BNEQ    4$                              ;
                                                59  D7 0008B          DECL    LVE                             ; 1016
                                                58  D7 0008D          DECL    LEN                             ; 1017
        01      A7      02      A7              59  28 0008F          MOVC3   LVE, 2(P), 1(P)                 ; 1018
                                                E9  11 00095          BRB     3$                              ; 1013
                                01              59  D1 00097 4$:      CMPL    LVE, #1                         ; 1024
                                                0B  15 0009A          BLEQ    5$                              ;
                                02              59  D1 0009C          CMPL    LVE, #2                         ; 1025
                                                09  12 0009F          BNEQ    6$                              ;
                                30      01      A7  91 000A1          CMPB    1(P), #48                       ;
                                                03  12 000A5          BNEQ    6$                              ;
                                58              59  C2 000A7 5$:      SUBL2   LVE, LEN                        ; 1027
                        FD34    CA              58  B0 000AA 6$:      MOVW    LEN, TMP_DESC                   ; 1028
                        FD38    CA              56  D0 000AF          MOVL    ADR, TMP_DESC+4                 ; 1029
                                        FD34    CA  9F 000B4          PUSHAB  TMP_DESC                        ; 1030
                                          08    AC  DD 000B8          PUSHL   OUT_DESC                        ;
                00000000G       00              02  FB 000BB          CALLS   #2, STR$COPY_DX                 ;
                                                04  00C2             RET                                     ; 1033

; Routine Size:  195 bytes,    Routine Base:  $CODE$ + 068C
```

```
;1043     1034  1 ROUTINE exclude_file =                    ! See if the file should be excluded
;1044     1035  1
;1045     1036  1 !++
;1046     1037  1 ! Functional description
;1047     1038  1 !
;1048     1039  1 !       This routine checks to see if the current resultant filename in
;1049     1040  1 !       nam_block matches any of the filenames in the exclude list
;1050     1041  1 !
;1051     1042  1 ! Calling sequence
;1052     1043  1 !
;1053     1044  1 !       exclude_file()
;1054     1045  1 !
;1055     1046  1 ! Input parameters
;1056     1047  1 !
;1057     1048  1 !       none
;1058     1049  1 !
;1059     1050  1 ! Implicit inputs
;1060     1051  1 !
;1061     1052  1 !       nam_block       - The NAMe block containing the resultant string
;1062     1053  1 !       exclude_desc    - Array of filenames to exclude
;1063     1054  1 !       cnt_exclude     - Count of items in use in the array
;1064     1055  1 !
;1065     1056  1 ! Output parameters
;1066     1057  1 !
;1067     1058  1 !       none
;1068     1059  1 !
;1069     1060  1 ! Implicit outputs
;1070     1061  1 !
;1071     1062  1 !       none
;1072     1063  1 !
;1073     1064  1 ! Routine value
;1074     1065  1 !
;1075     1066  1 !       TRUE    - The filename matches an item in the exclude list
;1076     1067  1 !       FALSE   - The filename does not match any item
;1077     1068  1 !
;1078     1069  1 ! Side effects
;1079     1070  1 !
;1080     1071  1 !       none
;1081     1072  1 !
;1082     1073  1 !--
;1083     1074  1
;1084     1075  2 BEGIN
;1085     1076  2
;1086     1077  2 LOCAL
;1087     1078  2     p,                  ! A local character pointer
;1088     1079  2     lenf,               ! Length of file name from nam_block
;1089     1080  2     adrf,               ! Address of    ''
;1090     1081  2     lent,               ! Length of target name from exclude list
;1091     1082  2     adrt;               ! Address of    ''
;1092     1083  2
;1093     1084  2 IF .cnt_exclude EQL 0 THEN RETURN false;! Don't bother doing anything more
;1094     1085  2
;1095     1086  2 !
;1096     1087  2 ! Remove device and directory from the result name
;1097     1088  2 !
;1098     1089  2 lenf = .nam_block [nam$b_rsl];          ! Get the res length
;1099     1090  2 adrf = .nam_block [nam$l_rsa];          ! Get the result address
```

```
;  1100          1091  2
;  1101          1092  2 p = locate_filename (.lenf, .adrf);      ! Look for end of directory
;  1102          1093  2 lenf = .lenf - (.p - .adrf);             ! Adjust the length
;  1103          1094  2 adrf = .p;                               ! Adjust the pointer
;  1104          1095  2
;  1105          1096  2 !
;  1106          1097  2 ! Now compare the file name against each file in the exclude list
;  1107          1098  2 !
;  1108          1099  2 INCR idx FROM 0 TO .cnt_exclude-1        ! No loop if it equals zero
;  1109          1100  2 DO
;  1110          1101  2     BEGIN
;  1111          1102  3     !
;  1112          1103  3     ! If the version is not explicitly given in the target, then remove
;  1113          1104  3     ! the version from both the input filename and the target
;  1114          1105  3     !
;  1115          1106  3     lent = .exclude_desc [.idx, dsc$w_length];  ! Length of target
;  1116          1107  3     adrt = .exclude_desc [.idx, dsc$a_pointer]; ! Address of target
;  1117          1108  3
;  1118          1109  3     p = locate_version (.lent, .adrt);          ! Look for version in target
;  1119          1110  3     IF .p EQL 0
;  1120          1111  3     THEN                            ! No version in target, strip from filename
;  1121          1112  4         BEGIN
;  1122          1113  4         p = locate_version (.lenf, .adrf);      ! Look for version in filename
;  1123          1114  4         IF .p NEQ 0                             ! If present
;  1124          1115  4         THEN
;  1125          1116  4             lenf = .p - .adrf;                  ! Set length without version
;  1126          1117  4         END;
;  1127          1118  3
;  1128          1119  3     IF match_file (.lenf, .adrf, .lent, .adrt)
;  1129          1120  3     THEN
;  1130          1121  3         RETURN true;                    ! We got a match, exclude this file
;  1131          1122  2     END;
;  1132          1123  2
;  1133          1124  2 RETURN false;                           ! Never matched, include this file
;  1134          1125  2
;  1135          1126  1 END;


                                        01FC 00000 EXCLUDE_FILE:
                                                   .WORD    Save R2,R3,R4,R5,R6,R7,R8          ; 1034
                                0000'   CF  D5 00002          TSTL     CNT_EXCLUDE            ; 1084
                                        72  13 00006          BEQL     4$                     ;
                            54  0000'   CF  9A 00008          MOVZBL   NAM_BLOCK+3, LENF      ; 1089
                            52  0000'   CF  D0 0000D          MOVL     NAM_BLOCK+4, ADRF      ; 1090
                                        52  DD 00012          PUSHL    ADRF                   ; 1092
                                        54  DD 00014          PUSHL    LENF                   ;
                        0000V   CF          02  FB 00016      CALLS    #2, LOCATE_FILENAME    ;
                            55              50  D0 0001B      MOVL     R0, P                  ;
                    50      52              55  C3 0001E      SUBL3    P, ADRF, R0            ; 1093
                            54              50  C0 00022      ADDL2    R0, LENF               ;
                            52              55  D0 00025      MOVL     P, ADRF                ; 1094
                            58  0000'   CF  D0 00028          MOVL     CNT_EXCLUDE, R8        ; 1099
                            53              01  CE 0002D      MNEGL    #1, IDX                ; 1119
                                            44  11 00030      BRB      3$                     ;
```

```
                                0000'CF43   7F  00032 1$:     PUSHAQ  EXCLUDE_DESC[IDX]                    ; 1106
                        57          9E   3C  00037          MOVZWL  @(SP)+, LENT
                                0000'CF43   7F  0003A          PUSHAQ  EXCLUDE_DESC+4[IDX]                 ; 1107
                        56          9E   D0  0003F          MOVL    @(SP)+, ADRT
                                    56   DD  00042          PUSHL   ADRT                                ; 1109
                                    57   DD  00044          PUSHL   LENT
                  0000V CF          02   FB  00046          CALLS   #2, LOCATE_VERSION
                        55          50   D0  0004B          MOVL    R0, P
                                    12   12  0004E          BNEQ    2$                                  ; 1110
                                    52   DD  00050          PUSHL   ADRF                                ; 1113
                                    54   DD  00052          PUSHL   LENF
                  0000V CF          02   FB  00054          CALLS   #2, LOCATE_VERSION
                        55          50   D0  00059          MOVL    R0, P
                                    04   13  0005C          BEQL    2$                                  ; 1114
                54                  52   C3  0005E          SUBL3   ADRF, P, LENF                       ; 1116
                                    56   DD  00062 2$:      PUSHL   ADRT                                ; 1119
                                0084 8F  BB  00064          PUSHR   #^M<R2,R7>
                                    54   DD  00068          PUSHL   LENF
                  0000V CF          04   FB  0006A          CALLS   #4, MATCH_FILE
                        04          50   E9  0006F          BLBC    R0, 3$
                        50          01   D0  00072          MOVL    #1, R0                              ; 1121
                                        04  00075          RET
                B8          53          58  F2  00076 3$:      AOBLSS  R8, IDX, 1$                     ; 1099
                                        50  D4  0007A 4$:      CLRL    R0                              ; 1126
                                        04  0007C          RET                                         ;

; Routine Size:  125 bytes,    Routine Base:  $CODE$ + 074F
```

```
;  1137     1127  1 ROUTINE locate_version (len, adr) =      ! Find version in filename
;  1138     1128  1
;  1139     1129  1 !++
;  1140     1130  1 ! Functional description
;  1141     1131  1 !
;  1142     1132  1 !       This routine finds the version number part of a file specification.
;  1143     1133  1 !       Both the standard syntax of NAME.TYP;1 and the alternate syntax of
;  1144     1134  1 !       NAME.TYP.1 are allowed.
;  1145     1135  1 !
;  1146     1136  1 ! Calling sequence
;  1147     1137  1 !
;  1148     1138  1 !       locate_version (.str [dsc$w_length], .str [dsc$a_pointer])
;  1149     1139  1 !
;  1150     1140  1 ! Input parameters
;  1151     1141  1 !
;  1152     1142  1 !       len - Length of filename string
;  1153     1143  1 !       adr - Address of filename string
;  1154     1144  1 !
;  1155     1145  1 ! Output parameters
;  1156     1146  1 !
;  1157     1147  1 !       none
;  1158     1148  1 !
;  1159     1149  1 ! Implicit inputs/outputs
;  1160     1150  1 !
;  1161     1151  1 !       none
;  1162     1152  1 !
;  1163     1153  1 ! Routine value
;  1164     1154  1 !
;  1165     1155  1 !       Address of the character delimiting the version number if found,
;  1166     1156  1 !       otherwise 0
;  1167     1157  1 !--
;  1168     1158  2 BEGIN
;  1169     1159  2
;  1170     1160  2 LOCAL
;  1171     1161  2     l,                              ! Remaining length for alternate syntax
;  1172     1162  2     p;                              ! Local character pointer
;  1173     1163  2
;  1174     1164  2 p = CH$FIND_CH (.len, .adr, ';');       ! Look for version in target
;  1175     1165  2 IF .p EQL 0
;  1176     1166  2 THEN
;  1177     1167  3     BEGIN
;  1178     1168  3     p = CH$FIND_CH (.len, .adr, '.');   ! Check for alternate syntax
;  1179     1169  3     IF .p EQL 0 THEN RETURN false;      ! No version
;  1180     1170  3     p = .p + 1;                         ! Bump past the dot
;  1181     1171  3     l = .len - (.p - .adr);             ! Length of remainder
;  1182     1172  3     IF .l LEQ 0 THEN RETURN false;      ! It was 'NAME.', no version
;  1183     1173  3     p = CH$FIND_CH (.l, .p, '.');       ! Look for second dot
;  1184     1174  2     END;
;  1185     1175  2
;  1186     1176  2 RETURN .p;                              ! Done, it is zero or pointer
;  1187     1177  2
;  1188     1178  1 END;
```

```
                        0000 00000 LOCATE_VERSION:
                                           .WORD   Save nothing                    ; 1127
          08   BC      04   AC             3B  3A 00002          LOCC    #59, LEN, @ADR          ; 1164
                                           02  12 00008          BNEQ    1$
                                           51  D4 0000A          CLRL    R1
                                           51  D5 0000C 1$:       TSTL    P                       ; 1165
                                           23  12 0000E          BNEQ    3$
          08   BC      04   AC             2E  3A 00010          LOCC    #46, LEN, @ADR          ; 1168
                                           02  12 00016          BNEQ    2$
                                           51  D4 00018          CLRL    R1
                                           51  D5 0001A 2$:       TSTL    P                       ; 1169
                                           19  13 0001C          BEQL    4$
                                           51  D6 0001E          INCL    P                       ; 1170
               50   08   AC                51  C3 00020          SUBL3   P, ADR, R0              ; 1171
                         50        04      AC  C0 00025          ADDL2   LEN, L
                                           0C  15 00029          BLEQ    4$                      ; 1172
               61        50                2E  3A 0002B          LOCC    #46, L, (P)             ; 1173
                                           02  12 0002F          BNEQ    3$
                                           51  D4 00031          CLRL    R1
                         50                51  D0 00033 3$:       MOVL    P, R0                   ; 1176
                                           04  00036          RET
                                           50  D4 00037 4$:       CLRL    R0                      ; 1178
                                           04  00039          RET

; Routine Size:  58 bytes,    Routine Base:  $CODE$ + 07CC
```

```
;  1190     1179  1 ROUTINE locate_filename (len, adr) =    ! Find version in filename
;  1191     1180  1
;  1192     1181  1 !++
;  1193     1182  1 ! Functional description
;  1194     1183  1 !
;  1195     1184  1 !       This routine finds the filename part of a file specification.
;  1196     1185  1 !       Node, device and directory are removed from the filespec.
;  1197     1186  1 !       Both the standard syntax of [DIR]NAME.TYP and the alternate
;  1198     1187  1 !       syntax of <DIR>NAME.TYP are allowed.
;  1199     1188  1 !
;  1200     1189  1 ! Calling sequence
;  1201     1190  1 !
;  1202     1191  1 !       locate_filename (.str [dsc$w_length], .str [dsc$a_pointer])
;  1203     1192  1 !
;  1204     1193  1 ! Input parameters
;  1205     1194  1 !
;  1206     1195  1 !       len - Length of filename string
;  1207     1196  1 !       adr - Address of filename string
;  1208     1197  1 !
;  1209     1198  1 ! Output parameters
;  1210     1199  1 !
;  1211     1200  1 !       none
;  1212     1201  1 !
;  1213     1202  1 ! Implicit inputs/outputs
;  1214     1203  1 !
;  1215     1204  1 !       none
;  1216     1205  1 !
;  1217     1206  1 ! Routine value
;  1218     1207  1 !
;  1219     1208  1 !       Address of the first character of the filename.  If no node, device
;  1220     1209  1 !       or directory is present, then .adr will be returned
;  1221     1210  1 !--
;  1222     1211  2 BEGIN
;  1223     1212  2
;  1224     1213  2 LOCAL
;  1225     1214  2     retval,                         ! Address to return
;  1226     1215  2     p;                              ! Local character pointer
;  1227     1216  2
;  1228     1217  2 retval = .adr;                              ! Assume nothing to remove
;  1229     1218  2
;  1230     1219  2 p = CH$FIND_CH (.len, .adr, ']');           ! Look for end of directory
;  1231     1220  2 IF .p NEQ 0
;  1232     1221  2 THEN
;  1233     1222  2     retval = .p + 1                        ! Move the pointer past the ']'
;  1234     1223  2 ELSE
;  1235     1224  3     BEGIN
;  1236     1225  3     p = CH$FIND_CH (.len, .adr, '>'); ! Alternate syntax for directory
;  1237     1226  3     IF .p NEQ 0
;  1238     1227  3     THEN
;  1239     1228  3         retval = .p + 1                    ! Move the pointer past the '>'
;  1240     1229  3     ELSE
;  1241     1230  4         BEGIN
;  1242     1231  4         LOCAL
;  1243     1232  4             l,
;  1244     1233  4             a;
;  1245     1234  4         l = .len;                          ! Get a temporary for the length
;  1246     1235  4         a = .adr;                          !  and the address
```

```
;  1247     1236  4          WHILE 1                          ! Loop and remove colons
;  1248     1237  4          DO
;  1249     1238  5              BEGIN
;  1250     1239  5              p = CH$FIND_CH (.l, .a, ':');
;  1251     1240  5              IF .p EQL 0 THEN EXITLOOP;
;  1252     1241  5              l = .l - (.p - .a + 1);      ! Adjust the length
;  1253     1242  5              a = .p + 1;                  ! Move the pointer past the ':'
;  1254     1243  4              END;
;  1255     1244  4          retval = .a;
;  1256     1245  3          END;
;  1257     1246  2      END;
;  1258     1247  2
;  1259     1248  2 RETURN .retval;                           ! Pass the address back
;  1260     1249  2
;  1261     1250  1 END;


                                        001C 00000 LOCATE_FILENAME:
                                                   .WORD   Save R2,R3,R4                    ; 1179
                        54      08   AC  D0 00002          MOVL    ADR, RETVAL             ; 1217
              08  BC        04  AC      5D  8F  3A 00006    LOCC    #93, LEN, @ADR          ; 1219
                                        02  12 0000D       BNEQ    1$                      ;
                                        51  D4 0000F       CLRL    R1                      ;
                        50              51  D0 00011 1$:   MOVL    R1, P                   ;
                                        0F  12 00014       BNEQ    3$                      ; 1220
              08  BC        04  AC      3E  3A 00016       LOCC    #62, LEN, @ADR          ; 1225
                                        02  12 0001C       BNEQ    2$                      ;
                                        51  D4 0001E       CLRL    R1                      ;
                        50              51  D0 00020 2$:   MOVL    R1, P                   ;
                                        06  13 00023       BEQL    4$                      ; 1226
                        54      01      A0  9E 00025 3$:   MOVAB   1(R0), RETVAL           ; 1228
                                        26  11 00029       BRB     8$                      ;
                        52      04  AC  7D 0002B 4$:      MOVQ    LEN, L                  ; 1234
                  63    52              3A  ?? 0002F 5$:   LOCC    #58, L, (A)             ; 1239
                                        0?  12 00033       BNEQ    6$                      ;
                                        51  D4 00035       CLRL    R1                      ;
                        50              51  D0 00037 6$:   MOVL    R1, P                   ;
                                        12  13 0003A       BEQL    7$                      ; 1240
                  51    50              53  C3 0003C       SUBL3   A, P, R1                ; 1241
                  51    52              51  C3 00040       SUBL3   R1, L, R1               ;
                        52      FF      A1  9E 00044       MOVAB   -1(R1), L               ;
                        53      01      A0  9E 00048       MOVAB   1(R0), A                ; 1242
                                        E1  11 0004C       BRB     5$                      ; 1236
                        54              53  D0 0004E 7$:   MOVL    A, RETVAL               ; 1244
                        50              54  D0 00051 8$:   MOVL    RETVAL, R0              ; 1248
                                            04 00054       RET                             ; 1250

; Routine Size:  85 bytes,    Routine Base:  $CODE$ + 0806
```

```
;  1263     1251  1 ROUTINE match_file (lenf, adrf, lent, adrt) =   ! Find version in filename
;  1264     1252  1
;  1265     1253  1 !++
;  1266     1254  1 ! Functional description
;  1267     1255  1 !
;  1268     1256  1 !       This routine compares a filename (result string) with a target
;  1269     1257  1 !       filename.  The target filename may contain the wildcard characters
;  1270     1258  1 !       "*" and "%".
;  1271     1259  1 !
;  1272     1260  1 ! Calling sequence
;  1273     1261  1 !
;  1274     1262  1 !       match_file (.lenf, .adrf, .lent, .adrt)
;  1275     1263  1 !
;  1276     1264  1 ! Input parameters
;  1277     1265  1 !
;  1278     1266  1 !       lenf - Length of filename string
;  1279     1267  1 !       adrf - Address of filename string
;  1280     1268  1 !       lent - Length of target filename string
;  1281     1269  1 !       adrt - Address of target filename string
;  1282     1270  1 !
;  1283     1271  1 ! Output parameters
;  1284     1272  1 !
;  1285     1273  1 !       none
;  1286     1274  1 !
;  1287     1275  1 ! Implicit inputs/outputs
;  1288     1276  1 !
;  1289     1277  1 !       none
;  1290     1278  1 !
;  1291     1279  1 ! Routine value
;  1292     1280  1 !
;  1293     1281  1 !       TRUE (1) if the target name includes the filename, FALSE (0)
;  1294     1282  1 !       otherwise
;  1295     1283  1 !--
;  1296     1284  2 BEGIN
;  1297     1285  2
;  1298     1286  2 LOCAL
;  1299     1287  2     cht,                       ! The character from target
;  1300     1288  2     chf;                       ! The character from the filename
;  1301     1289  2
;  1302     1290  2 DECR k FROM .lent-1 TO 0
;  1303     1291  2 DO
;  1304     1292  3     BEGIN
;  1305     1293  3     lent = .lent-1;                    ! Decrement target length
;  1306     1294  3     IF .lent LSS 0 THEN SIGNAL_STOP (SS$_BADPARAM);     ! Temporary check
;  1307     1295  3     cht = CH$RCHAR_A(adrt);            ! Fetch char and bump pointer
;  1308     1296  3
;  1309     1297  3     IF .cht EQL '*'            ! Found a wildcard in target
;  1310     1298  3     THEN
;  1311     1299  4         BEGIN
;  1312     1300  4         IF .lent EQL 0         ! Wildcard at end of target string
;  1313     1301  4         THEN                   !  matches everything, return a
;  1314     1302  4             RETURN true;       !  match
;  1315     1303  4         DECR i FROM .lenf-1 TO 0       ! Look through rest of filename
;  1316     1304  4         DO
;  1317     1305  5             BEGIN
;  1318     1306  5             IF match_file (.lenf, .adrf, .lent, .adrt)  ! Recursively
;  1319     1307  5             THEN                       !  examine rest of filename from this
```

```
;  1320          1308  5                RETURN true;     !  point in target, return true if found
;  1321          1309  5            lenf = .lenf-1;      ! Advance one character in the filename
;  1322          1310  5            adrf = .adrf+1;      !  and repeat recursive call
;  1323          1311  4            END;
;  1324          1312  4        RETURN false;            ! We did not match from wildcard
;  1325          1313  4        END
;  1326          1314  3    ELSE                         ! No wildcard in target
;  1327          1315  4        BEGIN
;  1328          1316  4        lenf = .lenf-1;                  ! Decrement the input string
;  1329          1317  4        IF .lenf LSS 0                   ! If we have exhausted the
;  1330          1318  4        THEN                             !  filename string then we did not
;  1331          1319  4            RETURN false;                !  match and we can return false
;  1332          1320  4        chf = CH$RCHAR_A (adrf);         ! Get filename char and bump pointer
;  1333          1321  5        IF NOT (                    ! If none of the successful tests
;  1334          1322  6                (.cht EQL .chf)     ! Did we match?
;  1335          1323  6                OR (.cht EQL '%')   ! Single character wildcard
;  1336          1324  6                OR ((.chf EQL ';') AND  ! See if the alternate syntax is
;  1337          1325  6                    (.cht EQL '.')    ) !  in target (assume RMS gives ';')
;  1338          1326  5               )
;  1339          1327  4        THEN                        ! We did not match in any way, therefore
;  1340          1328  4            RETURN false;           !  we can return false from here.
;  1341          1329  3        END;
;  1342          1330  2    END;
;  1343          1331  2
;  1344          1332  2 IF .lenf EQL 0 THEN RETURN true;          ! Matched so far, filename exhausted
;  1345          1333  2
;  1346          1334  2 RETURN false;                    ! More characters in filename, no match
;  1347          1335  2
;  1348          1336  1 END;


                                007C 00000 MATCH_FILE:
                                           .WORD   Save R2,R3,R4,R5,R6                    ;  1251
                55              0C   AC    D0 00002         MOVL    LENT, K                ;  1290
                                     63    11 00006         BRB     6$                     :
                                0C   AC    D7 00008 1$:     DECL    LENT                   ;  1293
                53              0C   AC    D0 0000B         MOVL    LENT, R3               ;  1294
                                     09    18 0000F         BGEQ    2$                     :
                                     14    DD 00011         PUSHL   #20                    :
00000000G       00                   01    FB 00013         CALLS   #1, LIB$STOP           :
                54              10   BC    9A 0001A 2$:     MOVZBL  @ADRT, CHT             ;  1295
                                10   AC    D6 0001E         INCL    ADRT                   :
                2A                   54    D1 00021         CMPL    CHT, #42               ;  1297
                                     25    12 00024         BNEQ    5$                     :
                                     53    D5 00026         TSTL    R3                     ;  1300
                                     49    13 00028         BEQL    7$                     :
                52              04   AC    D0 0002A         MOVL    LENF, I                ;  1306
                                     16    11 0002E         BRB     4$                     :
                                10   AC    DD 00030 3$:     PUSHL   ADRT                   :
                                     53    DD 00033         PUSHL   R3                     :
                7E              04   AC    7D 00035         MOVQ    LENF, -(SP)            :
      C3        AF                   04    FB 00039         CALLS   #4, MATCH_FILE         :
                33                   50    E8 0003D         BLBS    R0, 7$                 :
                                04   AC    D7 00040         DECL    LENF                   ;  1309
```

```
                    08   AC  D6 00043          INCL    ADRF                                    ; 1310
                E7       52  F4 00046 4$:      SOBGEQ  I, 3$                                   ; 1303
                         2C  11 00049          BRB     8$                                      ; 1312
                    04   AC  D7 0004B 5$:      DECL    LENF                                    ; 1316
                         27  19 0004E          BLSS    8$                                      ; 1317
                56  08   BC  9A 00050          MOVZBL  @ADRF, CHF                              ; 1320
                    08   AC  D6 00054          INCL    ADRF                                    ;
                56       54  D1 00057          CMPL    CHT, CHF                                ; 1322
                         0F  13 0005A          BEQL    6$                                      ;
                25       54  D1 0005C          CMPL    CHT, #37                                ; 1323
                         0A  13 0005F          BEQL    6$                                      ;
                3B       56  D1 00061          CMPL    CHF, #59                                ; 1324
                         11  12 00064          BNEQ    8$                                      ;
                2F       54  D1 00066          CMPL    CHT, #46                                ; 1325
                         0C  12 00069          BNEQ    8$                                      ;
                9A       55  F4 0006B 6$:      SOBGEQ  K, 1$                                   ; 1290
                    04   AC  D5 0006E          TSTL    LENF                                    ; 1332
                         04  12 00071          BNEQ    8$                                      ;
                50       01  D0 00073 7$:      MOVL    #1, R0                                  ;
                             04 00076          RET                                             ;
                         50  D4 00077 8$:      CLRL    R0                                      ; 1336
                             04 00079          RET                                             ;

; Routine Size:  122 bytes,    Routine Base:  $CODE$ + 085B
```

```
;  1350     1337  1 ROUTINE search_one_file : NOVALUE =     ! Open the file
;  1351     1338  1
;  1352     1339  1 !++
;  1353     1340  1 ! Functional description
;  1354     1341  1 !
;  1355     1342  1 !       This routine opens one file as described by the fab.  Control is then
;  1356     1343  1 !       passed to one of the routines which look for one of the search strings
;  1357     1344  1 !       and output any occurrences.
;  1358     1345  1 !
;  1359     1346  1 ! Calling sequence
;  1360     1347  1 !
;  1361     1348  1 !       search_one_file
;  1362     1349  1 !
;  1363     1350  1 ! Input parameters
;  1364     1351  1 !
;  1365     1352  1 !       none
;  1366     1353  1 !
;  1367     1354  1 ! Implicit inputs
;  1368     1355  1 !
;  1369     1356  1 !       fab             - The FAB  for the file specification
;  1370     1357  1 !       nam             - The related NAMe block
;  1371     1358  1 !       rab             - The RAB for the stream
;  1372     1359  1 !
;  1373     1360  1 ! Output parameters
;  1374     1361  1 !
;  1375     1362  1 !       none
;  1376     1363  1 !
;  1377     1364  1 ! Implicit outputs
;  1378     1365  1 !
;  1379     1366  1 !       none
;  1380     1367  1 !
;  1381     1368  1 ! Routine value
;  1382     1369  1 !
;  1383     1370  1 !       novalue
;  1384     1371  1 !
;  1385     1372  1 ! Side effects
;  1386     1373  1 !
;  1387     1374  1 !       Errors are signaled. Messages are put to SYS$OUTPUT.  Global variables
;  1388     1375  1 !       are updated.
;  1389     1376  1 !
;  1390     1377  1 !--
;  1391     1378  1
;  1392     1379  2 BEGIN
;  1393     1380  2
;  1394     1381  2 LOCAL
;  1395     1382  2     status;
;  1396     1383  2
;  1397     1384  3 IF NOT (status = $OPEN(FAB = fab))               ! Open the file
;  1398     1385  2 THEN
;  1399     1386  3         BEGIN
;  1400     1387  3         file_error (srh$_openin, .status, fab, .fab [fab$l_stv]);       ! Signal the error
;  1401     1388  3         RETURN;                                   ! Exit
;  1402     1389  2         END;
;  1403     1390  2
;  1404     1391  3 IF NOT (status = $CONNECT(RAB = rab))            ! Connect the rab
;  1405     1392  2 THEN
;  1406     1393  3         BEGIN
```

```
;  1407          1394  3          $CLOSE(FAB = fab);                          ! Close the file
;  1408          1395  3          file_error (srh$_openin, .status, fab, .rab [rab$l_stv]);       ! Signal the error
;  1409          1396  3          RETURN                                      ! Exit
;  1410          1397  2          END;
;  1411          1398  2
;  1412          1399  2 wdw_pre_buff = 0;                                    . Initialize the display window
;  1413          1400  2 wdw_pre_cntr = 0;
;  1414          1401  2 wdw_sub_cntr = 0;
;  1415          1402  2 fil_linenum = 0;
;  1416          1403  2 fil_totmat = 0;
;  1417          1404  2 fil_found = false;
;  1418          1405  2 new_file = true;
;  1419          1406  2
;  1420          1407  2 !
;  1421          1408  2 ! Search the opened file
;  1422          1409  2 !
;  1423          1410  2 std_search_file();
;  1424          1411  2
;  1425        L 1412  2 %IF switch_statistics
;  1426          1413  2 %THEN
;  1427          1414  2     stat_totfil = .stat_totfil+1;       ! Bump the count of files
;  1428          1415  2 %FI
;  1429          1416  2
;  1430          1417  2 IF .qua_log
;  1431          1418  2 THEN
;  1432          1419  3     BEGIN
;  1433          1420  3     tmp_desc [dsc$w_length] = .nam_block [nam$b_rsl];   ! Filename length
;  1434          1421  3     tmp_desc [dsc$a_pointer] = .nam_block [nam$l_rsa];  ! Filename address
;  1435          1422  3     IF .fil_found
;  1436          1423  3     THEN
;  1437          1424  4         BEGIN
;  1438          1425  4         LOCAL
;  1439          1426  4             m_sp;                                   ! Pointer to plural string
;  1440          1427  4         IF .fil_totmat EQL 1
;  1441          1428  4         THEN
;  1442          1429  4             m_sp = null_str                         ! No plural
;  1443          1430  4         ELSE
;  1444          1431  4             m_sp = %ASCID 'es';                     ! For matches.
;  1445          1432  4         SIGNAL (srh$_matched, 4, tmp_desc, .fil_linenum, .fil_totmat, .m_sp);
;  1446          1433  4         END
;  1447          1434  3     ELSE
;  1448          1435  3         SIGNAL (srh$_nomatch, 2, tmp_desc, .fil_linenum);
;  1449          1436  2     END;
;  1450          1437  2
;  1451          1438  2 $close(FAB = fab);              ! Close the file
;  1452          1439  2
;  1453          1440  1 END;


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                00297   .BLKB   1
                                     00  00  73  65  00298 P.ABZ:  .ASCII  \es\<0><0>                                ;
                                           010E0002  0029C P.ABY:  .LONG   17694722                                  ;
                                           00000000' 002A0         .ADDRESS P.ABZ                                    ;
```

```
                                        .EXTRN  SYS$OPEN

                                        .PSECT  $CODE$,NOWRT,2

                                007C 00000 SEARCH_ONE_FILE:
                                           .WORD   Save R2,R3,R4,R5,R6                  ; 1337
                          56 00000000G  00  9E 00002       MOVAB   LIB$SIGNAL, R6       ;
                          55 00000000G  00  9E 00009       MOVAB   SYS$CLOSE, R5        ;
                          54      0000' CF  9E 00010       MOVAB   FIL_TOTMAT, R4       ;
                          53      0000' CF  9E 00015       MOVAB   FAB, R3              ;
                                        53  DD 0001A       PUSHL   R3                   ; 1384
                00000000G 00            01  FB 0001C       CALLS   #1, SYS$OPEN         ;
                          52            50  D0 00023       MOVL    R0, STATUS           ;
                          05            52  E8 00026       BLBS    STATUS, 1$           ;
                                     0C A3  DD 00029       PUSHL   FAB+12               ; 1387
                                        18  11 0002C       BRB     2$                   ;
                                     50 A3  9F 0002E 1$:    PUSHAB  RAB                  ; 1391
                00000000G 00            01  FB 00031       CALLS   #1, SYS$CONNECT      ;
                          52            50  D0 00038       MOVL    R0, STATUS           ;
                          16            52  E8 0003B       BLBS    STATUS, 3$           ;
                                        53  DD 0003E       PUSHL   R3                   ; 1394
                          65            01  FB 00040       CALLS   #1, SYS$CLOSE        ;
                                     5C A3  DD 00043       PUSHL   RAB+12               ; 1395
                                     0C BB  00046 2$:      PUSHR   #^M<R2,R3>           ;
                             00D71098 8F  DD 00048       PUSHL   #14094488            ;
                    C000V CF            04  FB 0004E       CALLS   #4, FILE_ERROR       ;
                                        04  00053       RET                          ; 1393
                                     E0 A4  D4 00054 3$:    CLRL    WDW_PRE_CNTR         ; 1400
                                     E4 A4  7C 00057       CLRQ    WDW_SUB_CNTR         ; 1401
                                        64  7C 0005A       CLRQ    FIL_TOTMAT           ; 1403
                                     15 A4  94 0005C       CLRB    FIL_FOUND            ; 1404
                              12 A4         01  90 0005F       MOVB    #1, NEW_FILE         ; 1405
                            0000V CF        00  FB 00063       CALLS   #0, STD_SEARCH_FILE  ; 1410
                                  0CEC C4   D6  00068       INCL    STAT_TOTFIL          ; 1414
                                47   0F A4   E9  0006C       BLBC    QUA_LOG, 7$          ; 1417
                              98 A3   A3 A3   9B  00070       MOVZBW  NAM_BLOCK+3, TMP_DESC ; 1420
                              9C A3   A4 A3   D0  00075       MOVL    NAM_BLOCK+4, TMP_DESC+4 ; 1421
                                28   15 A4   E9  0007A       BLBC    FIL_FOUND, 6$        ; 1422
                                01         64  D1  0007E       CMPL    FIL_TOTMAT, #1       ; 1427
                                           07  12  00081       BNEQ    4$                   ;
                                50      0000' CF  9E 00083       MOVAB   NULL_STR, M_SP       ; 1429
                                           05  11  00088       BRB     5$                   ;
                                50      0000' CF  9E 0008A 4$:    MOVAB   P.ABY, M_SP          ; 1431
                                           50  DD  0008F 5$:    PUSHL   M_SP                 ; 1432
                                           64  DD  00091       PUSHL   FIL_TOTMAT           ;
                                        04 A4  DD  00093       PUSHL   FIL_LINENUM          ;
                                        98 A3  9F  00096       PUSHAB  TMP_DESC             ;
                                           04  DD  00099       PUSHL   #4                   ;
                                00000000G  8F  DD  0009B       PUSHL   #SRH$_MATCHED        ;
                                66         06  FB  000A1       CALLS   #6, LIB$SIGNAL       ;
                                           11  11  000A4       BRB     7$                   ; 1422
                                        04 A4  DD  000A6 6$:    PUSHL   FIL_LINENUM          ; 1435
                                        98 A3  9F  000A9       PUSHAB  TMP_DESC             ;
                                           02  DD  000AC       PUSHL   #2                   ;
                                00000000G  8F  DD  000AE       PUSHL   #SRH$_NOMATCH        ;
                                66         04  FB  000B4       CALLS   #4, LIB$SIGNAL       ;
                                           53  DD  000B7 7$:    PUSHL   R3                   ; 1438
```

```
                                65          01  FB 000B9          CALLS   #1, SYS$CLOSE                                    ;
                                                04 000BC          RET                                                     ; 1440

; Routine Size:  189 bytes,    Routine Base:  $CODE$ + 08D5
```

```
;  1455     1441  1 ROUTINE get_next_record (len, adr)  : JSB_GETNEXREC =
;  1456     1442  2 BEGIN
;  1457     1443  2
;  1458     1444  2 !++
;  1459     1445  2 ! Description:
;  1460     1446  2 !
;  1461     1447  2 !       This routine gets the next record from the current input stream.
;  1462     1448  2 !       It returns the length and address of the re ord.  The routine
;  1463     1449  2 !       returns "true" if a record is present, "false" if no record is
;  1464     1450  2 !       found.  Calling routines should treat all "false" returns as
;  1465     1451  2 !       End of File.  If an actual error occurs, it will be signaled.
;  1466     1452  2 !--
;  1467     1453  2
;  1468     1454  2 OWN_DZRO
;  1469     1455  2     trunc;
;  1470     1456  2
;  1471     1457  2 LOCAL
;  1472     1458  2     get_recsiz,
;  1473     1459  2     status;
;  1474     1460  2
;  1475     1461  2 IF .new_file
;  1476     1462  2 THEN
;  1477     1463  3     BEGIN
;  1478     1464  3     new_file = false;
;  1479     1465  3     trunc = false;
;  1480     1466  2     END;
;  1481     1467  2
;  1482     1468  2 status = $GET(RAB = rab);                ! Get a record
;  1483     1469  2 get_recsiz = .rab [rab$w_rsz];           ! Copy length of record
;  1484     1470  2
;  1485     1471  2 IF (.status EQLU rms$_rtb) OR
;  1486     1472  3    (.status AND (.get_recsiz GTRU io_buff_sz))
;  1487     1473  2 THEN
;  1488     1474  3     BEGIN
;  1489     1475  3     get_recsiz = io_buff_sz;
;  1490     1476  3     IF NOT .trunc
;  1491     1477  3     THEN
;  1492     1478  4         BEGIN
;  1493     1479  4         trunc = true;
;  1494     1480  4         tmp_desc [dsc$w_length]  = .nam_block [nam$b_rsl];
;  1495     1481  4         tmp_desc [dsc$a_pointer] = .nam_block [nam$l_rsa];
;  1496     1482  4         SIGNAL (srh$_truncate,2,tmp_desc,io_buff_sz);
;  1497     1483  3         END;
;  1498     1484  3     END
;  1499     1485  2 ELSE
;  1500     1486  2     IF NOT .status                              ! If error
;  1501     1487  2     THEN
;  1502     1488  3         BEGIN
;  1503     1489  3         IF .status NEQ RMS$_EOF          ! If not end-of-file
;  1504     1490  3         THEN
;  1505     1491  3             file_error (srh$_readerr, .status, fab, .rab [rab$l_stv]);
;  1506     1492  3         IF .fil_linenum EQL 0
;  1507     1493  3         THEN
;  1508     1494  4             BEGIN
;  1509     1495  4             tmp_desc [dsc$w_length]  = .nam_block [nam$b_rsl];
;  1510     1496  4             tmp_desc [dsc$a_pointer] = .nam_block [nam$l_rsa];
;  1511     1497  4             SIGNAL (srh$_nullfile, 1, tmp_desc);
```

```
;  1512          1498  3             END;
;  1513          1499  3         RETURN false;                   ! and exit
;  1514          1500  2         END;
;  1515          1501  2
;  1516          1502  2 !
;  1517          1503  2 ! Bump the statistics info
;  1518          1504  2 !
;  1519          1505  2 fil_linenum = .fil_linenum + 1;         ! Bump the line (record) cntr
;  1520        L 1506  2 %IF switch_statistics
;  1521          1507  2 %THEN
;  1522          1508  2     stat_totrec = .stat_totrec + 1;             ! Bump the total too
;  1523          1509  2     stat_totchr = .stat_totchr + .get_recsiz;   ! Add the character count
;  1524          1510  2 %FI
;  1525          1511  2
;  1526          1512  2 IF .wdw_previous                ! Save the window if windowing is active
;  1527          1513  2 THEN
;  1528          1514  2     advance_window();
;  1529          1515  2
;  1530          1516  2 !
;  1531          1517  2 ! return the length and address and RFA of the new record
;  1532          1518  2 !
;  1533          1519  2 .len = .get_recsiz;
;  1534          1520  2 .adr = .rab [rab$l_rbf];
;  1535          1521  2
;  1536          1522  2 RETURN true;
;  1537          1523  2
;  1538          1524  1 END;


                                                        .PSECT  $OWN$_DZRO,NOEXE,2

                                                01347           .BLKB   1
                                                01348 TRUNC:    .BLKB   4

                                                        .EXTRN  SYS$GET

                                                        .PSECT  $CODE$,NOWRT,2

                08      0000'   CF   E9 00000 GET_NEXT_RECORD:
                                             BLBC    NEW_FILE, 1$                    ;  1461
                        0000'   CF   94 00005        CLRB    NEW_FILE               ;  1464
                        0000'   CF   D4 00009        CLRL    TRUNC                  ;  1465
                        0000'   CF   9F 0000D 1$:    PUSHAB  RAB                    ;  1468
00000000G       00              01   FB 00011        CALLS   #1, SYS$GET            ;
                52      0000'   CF   3C 00018        MOVZWL  RAB+34, GET_RECSIZ     ;  1469
000181A8        8F              50   D1 0001D        CMPL    STATUS, #98728         ;  1471
                                0C   13 00024        BEQL    2$                     ;
                43              50   E9 00026        BLBC    STATUS, 4$             ;  1472
00000800        8F              52   D1 00029        CMPL    GET_RECSIZ, #2048      ;
                                37   1B 00030        BLEQU   3$                     ;
                52      0800    8F   3C 00032 2$:    MOVZWL  #2048, GET_RECSIZ      ;  1475
                77      0000'   CF   E8 00037        BLBS    TRUNC, 6$              ;  1476
          0000' CF              01   D0 0003C        MOVL    #1, TRUNC              ;  1479
          0000' CF      0000'   CF   9B 00041        MOVZBW  NAM_BLOCK+3, TMP_DESC  ;  1480
          0000' CF      0000'   CF   D0 00048        MOVL    NAM_BLOCK+4, TMP_DESC+4;  1481
                7E      0800    8F   3C 0004F        MOVZWL  #2048, -(SP)           ;  1482
```

```
                          0000'  CF   9F 00054          PUSHAB  TMP_DESC                                    ;
                                 02   DD 00058          PUSHL   #2                                         ;
                     00000000G   8F   DD 0005A          PUSHL   #SRH$_TRUNCATE                             ;
000000J0G   00                   04   FB 00060          CALLS   #4, LIB$SIGNAL                             ;
                                 4A   11 00067          BRB     6$                                         ;  1471
                          47     50   E8 00069 3$:       BLBS    STATUS, 6$                                 ;  1486
0001827A    8F                   50   D1 0006C 4$:       CMPL    STATUS, #98938                             ;  1489
                                 15   13 00073          BEQL    5$                                         ;
                          0000'  CF   DD 00075          PUSHL   RAB+12                                     ;  1491
                          0000'  CF   9F 00079          PUSHAB  FAB                                        ;
                                 50   DD 0007D          PUSHL   STATUS                                     ;
                     00D710B0    8F   DD 0007F          PUSHL   #14094512                                  ;
          0000V      CF                04   FB 00085          CALLS   #4, FILE_ERROR                             ;
                          0000'  CF   D5 0008A 5$:       TSTL    FIL_LINENUM                                ;  1492
                                 46   12 0008E          BNEQ    8$                                         ;
          0000'      CF  0000'  CF   9B 00090          MOVZBW  NAM_BLOCK+3, TMP_DESC                      ;  1495
          0000'      CF  0000'  CF   D0 00097          MOVL    NAM_BLOCK+4, TMP_DESC+4                    ;  1496
                          0000'  CF   9F 0009E          PUSHAB  TMP_DESC                                   ;  1497
                                 01   DD 000A2          PUSHL   #1                                         ;
                     00000000G   8F   DD 000A4          PUSHL   #SRH$_NULLFILE                             ;
00000000G   00                   03   FB 000AA          CALLS   #3, LIB$SIGNAL                             ;
                                 23   11 000B1          BRB     8$                                         ;  1499
                          0000'  CF   D6 000B3 6$:       INCL    FIL_LINENUM                                ;  1505
                          0000'  CF   D6 000B7          INCL    STAT_TOTREC                                ;  1508
          0000'      CF                52   C0 000BB          ADDL2   GET_RECSIZ, STAT_TOTCHR                    ;  1509
                     05   0000'  CF   E9 000C0          BLBC    WDW_PREVIOUS, 7$                           ;  1512
          0000V      CF                00   FB 000C5          CALLS   #0, ADVANCE_WINDOW                         ;  1514
                     63                52   D0 000CA 7$:       MOVL    GET_RECSIZ, (LEN)                          ;  1519
                     64   0000'  CF   DC 000CD          MOVL    RAB+40, (ADR)                              ;  1520
                     50                01   D0 000D2          MOVL    #1, R0                                     ;  1522
                                            05 000D5          RSB                                                ;
                                       50   D4 000D6 8$:       CLRL    R0                                         ;  1524
                                            05 000D8          RSB                                                ;

; Routine Size:  217 bytes,    Routine Base:  $CODE$ + 0992
```

```
;  1540          1525  1 ROUTINE advance_window : NOVALUE =
;  1541          1526  2 BEGIN
;  1542          1527  2 !
;  1543          1528  2 ! Increment the number of records available for display
;  1544          1529  2 !
;  1545          1530  2 IF .wdw_pre_cntr LEQU .wdw_pre_recs
;  1546          1531  2 THEN
;  1547          1532  3     BEGIN
;  1548          1533  3     wdw_pre_cntr = .wdw_pre_cntr + 1;
;  1549          1534  3     wdw_ovfl = false;
;  1550          1535  3     END
;  1551          1536  2 ELSE                              ! Window can't be contiguous
;  1552          1537  2     wdw_ovfl = true;              ! with previous window.
;  1553          1538  2 !
;  1554          1539  2 ! Scroll the window to accomodate the next record, wraparound if needed
;  1555          1540  2 !
;  1556          1541  2 IF .wdw_pre_buff GTRU .wdw_pre_recs      ! if we have hit the end
;  1557          1542  2 THEN                                    ! then
;  1558          1543  2     wdw_pre_buff = 1                    !  reset to top of array
;  1559          1544  2 ELSE                                    ! otherwise
;  1560          1545  2     wdw_pre_buff = .wdw_pre_buff + 1;   !   advance to next position
;  1561          1546  2 !
;  1562          1547  2 ! Save the RFA in the previous record vector
;  1563          1548  2 !
;  1564          1549  2 CH$MOVE(rab$s_rfa, rab [rab$w_rfa], rfa_vec [.wdw_pre_buff-1,0,0,0,0]);
;  1565          1550  2
;  1566          1551  2 RETURN;
;  1567          1552  1 END;


                                                007C 00000 ADVANCE_WINDOW:
                                                           .WORD   Save R2,R3,R4,R5,R6                    ; 1525
                                    56    0000'  CF  9E 00002       MOVAB   WDW_PRE_BUFF, R6               ;
                                F0  A6      F8   A6  D1 00007       CMPL    WDW_PRE_CNTR, WDW_PRE_RECS     ; 1530
                                                08  1A 0000C       BGTRU   1$                             ;
                                              F8   A6  D6 0000E     INCL    WDW_PRE_CNTR                   ; 1533
                                              25   A6  94 00011     CLRB    WDW_OVFL                       ; 1534
                                                04  11 00014       BRB     2$                             ; 1530
                                25  A6                01  90 00016 1$:    MOVB    #1, WDW_OVFL           ; 1537
                                F0  A6                66  D1 0001A 2$:    CMPL    WDW_PRE_BUFF, WDW_PRE_RECS ; 1541
                                                05  1B 0001E       BLEQU   3$                             ;
                                    66              01  D0 00020       MOVL    #1, WDW_PRE_BUFF           ; 1543
                                                02  11 00023       BRB     4$                             ;
                                                66  D6 00025 3$:    INCL    WDW_PRE_BUFF                   ; 1545
                        50          66              06  C5 00027 4$:    MULL3   #6, WDW_PRE_BUFF, R0       ; 1549
                                    50      30  A6  C0 0002B       ADDL2   RFA_VEC, R0                    ;
                    FA  A0    0000'  CF            06  28 0002F       MOVC3   #6, RAB+16, -6(R0)         ;
                                                04  00036       RET                                    ; 1552

; Routine Size:  55 bytes,    Routine Base:  $CODE$ + 0A6B
```

```
;1569     1553  1 ROUTINE std_search_file : NOVALUE =                    ! Searches the file
;1570     1554  1
;1571     1555  1 !++
;1572     1556  1 ! Functional description
;1573     1557  1 !
;1574     1558  1 !       This routine searches one file (fab) for one of the search strings
;1575     1559  1 !       and outputs any occurrences.
;1576     1560  1 !
;1577     1561  1 ! Calling sequence
;1578     1562  1 !
;1579     1563  1 !       search_file
;1580     1564  1 !
;1581     1565  1 ! Input parameters
;1582     1566  1 !
;1583     1567  1 !       none
;1584     1568  1 !
;1585     1569  1 ! Implicit inputs
;1586     1570  1 !
;1587     1571  1 !       fab             - The FAB  for the file specification
;1588     1572  1 !       nam             - The related NAMe block
;1589     1573  1 !
;1590     1574  1 ! Output parameters
;1591     1575  1 !
;1592     1576  1 !       none
;1593     1577  1 !
;1594     1578  1 ! Implicit outputs
;1595     1579  1 !
;1596     1580  1 !       The filename and any occurrences of the string are put to
;1597     1581  1 !       SYS$OUTPUT.
;1598     1582  1 !
;1599     1583  1 ! Routine value
;1600     1584  1 !
;1601     1585  1 !       novalue
;1602     1586  1 !
;1603     1587  1 ! Side effects
;1604     1588  1 !
;1605     1589  1 !       Errors are signaled. Messages are put to SYS$OUTPUT.
;1606     1590  1 !
;1607     1591  1 !--
;1608     1592  1
;1609     1593  2 BEGIN
;1610     1594  2
;1611     1595  2 WHILE 1 DO                                       ! Main search loop
;1612     1596  3         BEGIN
;1613     1597  3         LOCAL
;1614     1598  3             rec_siz,            ! Length of input record
;1615     1599  3             rec_ptr,            ! Pointer to record
;1616     1600  3             status;
;1617     1601  3
;1618     1602  3         IF NOT get_next_record (rec_siz, rec_ptr)
;1619     1603  3         THEN
;1620     1604  3             EXITLOOP;
;1621     1605  3
;1622     1606  3         !
;1623     1607  3         ! See if record contains search string
;1624     1608  3         !
;1625     1609  3         IF std_match(.rec_siz, .rec_ptr)        ! See if string is present
```

```
;  1626     1610  3          THEN
;  1627     1611  4              BEGIN
;  1628     1612  4              IF .out_printing                        ! we know wdw_zero is off
;  1629     1613  4              THEN
;  1630     1614  5                  BEGIN
;  1631     1615  5                  IF NOT .fil_found                           ! If not already found
;  1632     1616  6                      AND (.nam_block [nam$v_wildcard] OR     ! and wildcard filespec
;  1633     1617  6                           .cnt_infile GTRU 1)                ! or list of files
;  1634     1618  5                      AND .qua_heading                        ! And heading requested
;  1635     1619  5                  THEN
;  1636     1620  6                      BEGIN       ! Print blank, filename, blank
;  1637     1621  6                      srh$put_output(0, 0);
;  1638     1622  6                      srh$put_output(30, stars_30);
;  1639     1623  6                      srh$put_output(.nam_block [nam$b_rsl], .nam_block [nam$l_rsa]);
;  1640     1624  6                      srh$put_output(0, 0);
;  1641     1625  6                      END
;  1642     1626  5                  ELSE
;  1643     1627  6                      BEGIN
;  1644     1628  6                      IF .fil_found                   ! If already found,
;  1645     1629  6                      THEN
;  1646     1630  7                          BEGIN
;  1647     1631  7                          IF      .wdw_frame          ! Separate the displays if the
;  1648     1632  7                              AND .wdw_ovfl           ! windows are discontiguous
;  1649     1633  7                              AND .qua_heading        ! And heading requested
;  1650     1634  7                          THEN
;  1651     1635  7                              srh$put_output(15, stars_30);
;  1652     1636  6                          END;
;  1653     1637  5                      END;
;  1654     1638  5                  !
;  1655     1639  5                  ! Put the previous records (if any) and the current record
;  1656     1640  5                  !
;  1657     1641  5                  IF .wdw_pre_cntr LEQ 1
;  1658     1642  5                  THEN
;  1659     1643  5                      ! pre_cntr LEQ 1 is our flag then the window EQL 1,
;  1660     1644  5                      ! just print the current record, allowing for reformat
;  1661     1645  5                      !
;  1662     1646  5                      format_n_put(.rec_siz, .rec_ptr, .fil_lirenum)
;  1663     1647  5                  ELSE
;  1664     1648  6                      BEGIN
;  1665     1649  6                      LOCAL
;  1666     1650  6                          status,
;  1667     1651  6                          idx;
;  1668     1652  6                      !
;  1669     1653  6                      ! Do a find by RFA to set to start of window
;  1670     1654  6                      !
;  1671     1655  6                      rab [rab$b_rac] = rab$c_rfa;        ! Set to random by RFA
;  1672     1656  7                      idx = (.wdw_pre_recs - .wdw_pre_cntr + .wdw_pre_buff + 1)
;  1673     1657  6                            MOD (.wdw_pre_recs + 1);
;  1674     1658  6                      CH$MOVE(rab$s_rfa, rfa_vec [.idx,0,0,0,0], rab [rab$w_rfa]);
;  1675     1659  7                      IF NOT (status = $FIND(RAB = rab)) ! Find the record
;  1676     1660  6                      THEN
;  1677     1661  7                          BEGIN
;  1678     1662  7                          SIGNAL_STOP (srh$_rfaerr, 2, .rfa_vec [.idx,0,0,32,0],
;  1679     1663  7                                       .rfa_vec [.idx,4,0,16,0], .status, .rab [rab$[_stv]);
;  1680     1664  7                          RETURN;
;  1681     1665  6                          END;
;  1682     1666  6                      rab [rab$b_rac] = rab$c_seq;        ! Back to sequential mode
```

```
;  1683        1667  6                !
;  1684        1668  6                ! Now do GETs to reread the records for the window
;  1685        1669  6                !
;  1686        1670  6                INCRU i FROM 1 TO .wdw_pre_cntr     ! Now put the records
;  1687        1671  6                DO
;  1688        1672  7                    BEGIN
;  1689        1673  7                    LOCAL
;  1690        1674  7                        actual_line,        ! Line number adjusted for wdw
;  1691        1675  7                        len,
;  1692        1676  7                        status;
;  1693        1677  7                    status = $GET(RAB = rab);
;  1694        1678  7                    len = .rab [rab$w_rsz]; ! Get actual length read
;  1695        1679  7                    IF (.status EQLU rms$_rtb) OR
;  1696        1680  8                       (.status AND (.len GTRU io_buff_sz))
;  1697        1681  7                    THEN
;  1698        1682  7                        len = io_buff_sz    ! Would have warned at 1st get
;  1699        1683  7                    ELSE
;  1700        1684  7                        IF NOT .status      ! If any other error
;  1701        1685  7                        THEN                ! signal and stop
;  1702        1686  8                            BEGIN
;  1703        1687  8                            SIGNAL_STOP (srh$_rfaerr, 2, .rfa_vec [.idx,0,0,32,0],
;  1704        1688  8                                .rfa_vec [.idx,4,0,16,0], .status, .rab [rab$l_stv]);
;  1705        1689  8                            RETURN;
;  1706        1690  7                            END;
;  1707        1691  7                    actual_line = .fil_linenum-(.wdw_pre_cntr-.i);
;  1708        1692  7                    format_n_put(.len, .rab [rab$l_rbf], .actual_line);
;  1709        1693  6                    END;
;  1710        1694  5                END;
;  1711        1695  5            wdw_pre_cntr = 0;               ! First half of window is done
;  1712        1696  5            wdw_sub_cntr = .wdw_sub_recs;
;  1713        1697  5            END
;  1714        1698  4        ELSE
;  1715        1699  5            BEGIN
;  1716        1700  5            IF .wdw_zero                    ! Print filename if /WIN=0
;  1717        1701  5            THEN
;  1718        1702  5                IF NOT .fil_found           ! Only need to do it once though
;  1719        1703  5                THEN
;  1720        1704  5                    srh$put_output(.nam_block [nam$b_rsl], .nam_block [nam$l_rsa]);
;  1721        1705  4                END;
;  1722        1706  4            fil_found = true;               ! Set match flag for this file
;  1723        1707  4            found_any = true;               ! Set global match flag
;  1724        1708  4            fil_totmat = .fil_totmat+1;     ! Count match for this file
;  1725        1709  4            END
;  1726        1710  3        ELSE                    ! We don't have a match
;  1727        1711  4            BEGIN               ! If we have subsequent lines to display
;  1728        1712  5            IF .out_printing AND (.wdw_sub_cntr GTRU 0)
;  1729        1713  4            THEN
;  1730        1714  5                BEGIN
;  1731        1715  5                format_n_put(.rec_siz, .rec_ptr, .fil_linenum);
;  1732        1716  5                wdw_pre_cntr = 0;       ! remember what we've shown.
;  1733        1717  5                wdw_sub_cntr = .wdw_sub_cntr - 1;
;  1734        1718  4                END;
;  1735        1719  4            END
;  1736        1720  2        END;
;  1737        1721  2 RETURN;
;  1738        1722  1 END;
```

```
                                        .EXTRN  SYS$FIND

                        OFFC 00000 STD_SEARCH_FILE:
                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11            ; 1553
5B      0000V   CF   9E 00002          MOVAB   FORMAT_N_PUT, R11
5A      0000V   CF   9E 00007          MOVAB   SRH$PUT_OUTPUT, R10
59      0000'   CF   9E 0000C          MOVAB   NAM_BLOCK+4, R9
58      0000'   CF   9E 00011          MOVAB   WDW_PRE_CNTR, R8
5E            08   C2 00016          SUBL2   #8, SP
54            6E   9E 00019 1$:       MOVAB   REC_PTR, R4                                     ; 1602
53        04  AE   9E 0001C          MOVAB   REC_SIZ, R3
          FECD  30 00020          BSBW    GET_NEXT_RECORD
01              50   E8 00023          BLBS    R0, 2$
                04 00026          RET
55            6E   D0 00027 2$:       MOVL    REC_PTR, R5                                     ; 1679
54        0'  AE   D0 0002A          MOVL    REC_SIZ, R4
          0000V 30 0002E          BSBW    STD_MATCH
03              50   E8 00031          BLBS    R0, 3$
              0132   31 00034          BRW     20$
03        29  A8   E8 00037 3$:       BLBS    OUT_PRINTING, 4$                                ; 1612
              010F   31 0003B          BRW     18$
2D        35  A8   E8 0003E 4$:       BLBS    FIL_FOUND, 7$                                   ; 1615
06        31  A9   E8 00042          BLBS    NAM_BLOCK+53, 5$                                ; 1616
01        0C  A8   D1 00046          CMPL    CNT_INFILE, #1                                  ; 1617
              1F   1B 0004A          BLEQU   6$
1B        31  A8   E9 0004C 5$:       BLBC    QUA_HEADING, 6$                                 ; 1618
              52   7C 00050          CLRQ    R2                                              ; 1621
              6A   16 00052          JSB     SRH$PUT_OUTPUT
          53  0000'  CF   9E 00054          MOVAB   STARS_30, R3                            ; 1622
          52         1E   D0 00059          MOVL    #30, R2
                     6A   16 0005C          JSB     SRH$PUT_OUTPUT
          53         69   D0 0005E          MOVL    NAM_BLOCK+4, R3                         ; 1623
          52      FF    A9   9A 00061          MOVZBL  NAM_BLOCK+3, R2
                     6A   16 00065          JSB     SRH$PUT_OUTPUT
                     52   7C 00067          CLRQ    R2                                      ; 1624
                     18   11 00069          BRB     8$
          16         35   A8   E9 0006B 6$:       BLBC    FIL_FOUND, 9$                   ; 1628
          12         2C   A8   E9 0006F 7$:       BLBC    WDW_FRAME, 9$                   ; 1631
          0E         2D   A8   E9 00073          BLBC    WDW_OVFL, 9$                    ; 1632
          0A         31   A8   E9 00077          BLBC    QUA_HEADING, 9$                 ; 1633
          53  0000'  CF   9E 0007B          MOVAB   STARS_30, R3                            ; 1635
          52         0F   D0 00080          MOVL    #15, R2
                     6A   16 00083 8$:       JSB     SRH$PUT_OUTPUT
          01         68   D1 00085 9$:       CMPL    WDW_PRE_CNTR, #1                        ; 1641
                     10   14 00088          BGTR    10$
          54         24   A8   D0 0008A          MOVL    FIL_LINENUM, R4                 ; 1646
          53         6E   D0 0008E          MOVL    REC_PTR, R3
          52      04    AE   D0 00091          MOVL    REC_SIZ, R2
                     6B   16 00095          JSB     FORMAT_N_PUT
                  00AA   31 00097          BRW     17$
              00CA   C9   02   90 0009A 10$:       MOVB    #2, RAB+30                      ; 1655
    51         F8  A8   68   C3 0009F          SUBL3   WDW_PRE_CNTR, WDW_PRE_RECS, R1          ; 1656
          51      08   A8   C0 000A4          ADDL2   WDW_PRE_BUFF, R1
    50         F8  A8   01   C1 000A8          ADDL3   #1, WDW_PRE_RECS, R0                    ; 1657
7E      01         51   01   7A 000AD          EMUL    #1, R1, #1, -(SP)
```

```
                50              50              8E              50  7B 000B2          EDIV    R0, (SP)+, IDX, IDX
                56              50              06              C5 000B7          MULL3   #6, IDX, R6                                 ;1658
                        00BC    C9      38 B846                 06  28 000BB      MOVC3   #6, @RFA_VEC[R6], RAB+16
                                                        00AC    C9  9F 000C3      PUSHAB  RAB                                         ;1659
                            00000000G   00                      01  FB 000C7      CALLS   #1, SYS$FIND
                                        3B                      50  E9 000CE      BLBC    STATUS, 14$
                                                        00CA    C9  94 000D1      CLRB    RAB+30                                      ;1666
                                        57                      68  D0 000D5      MOVL    WDW_PRE_CNTR, R7                            ;1670
                                        55                      01  D0 000D8      MOVL    #1, I
                                                                62  11 000DB      BRB     16$
                                                        00AC    C9  9F 000DD 11$:  PUSHAB  RAB                                         ;1677
                            00000000G   00                      01  FB 000E1      CALLS   #1, SYS$GET
                                        52              00CE    C9  3C 000E8      MOVZWL  RAB+34, LEN                                 ;1678
                            000181A8    8F                      50  D1 000ED      CMPL    STATUS, #98728                              ;1679
                                                                0C  13 000F4      BEQL    12$
                                        13                      50  E9 000F6      BLBC    STATUS, 14$                                 ;1680
                            00000800    8F                      52  D1 000F9      CMPL    LEN, #2048
                                                                07  1B 00100      BLEQU   13$
                                        52              0800    8F  3C 00102 12$:  MOVZWL  #2048, LEN                                  ;1682
                                                                24  11 00107      BRB     15$
                                        21                      50  E8 00109 13$:  BLBS    STATUS, 15$                                 ;1684
                                                        00B8    C9  DD 0010C 14$:  PUSHL   RAB+12                                      ;1688
                                                                50  DD 00110      PUSHL   STATUS
                50                      56              38      A8  C1 00112      ADDL3   RFA_VEC, R6, R0
                                        7E              04      A0  3C 00117      MOVZWL  4(R0), -(SP)
                                                                60  DD 0011B      PUSHL   (R0)                                        ;1687
                                                                02  DD 0011D      PUSHL   #2
                                                00000000G       8F  DD 0011F      PUSHL   #SRH$_RFAERR
                            00000000G   00                      06  FB 00125      CALLS   #6, LIB$STOP
                                                                04  0012C      RET                                                 ;1686
        50                      55                      68  C3 0012D 15$:  SUBL3   WDW_PRE_CNTR, I, R0                            ;1691
        54                      50              24      A8  C1 00131      ADDL3   FIL_LINENUM, R0, ACTUAL_LINE
                                53              00D4    C9  D0 00136      MOVL    RAB+40, R3                                  ;1692
                                                        6B  16 0013B      JSB     FORMAT_N_PUT
                                                        55  D6 0013D      INCL    I                                               ;1670
                                        57              55  D1 0013F 16$:  CMPL    I, R7
                                                        99  1B 00142      BLEQU   11$
                                                        68  D4 00144 17$:  CLRL    WDW_PRE_CNTR                                    ;1695
                                    04  A8              FC  A8  D0 00146      MOVL    WDW_SUB_RECS, WDW_SUB_CNTR                  ;1696
                                                        11  11 0014B      BRB     19$                                             ;1612
                                        0D              2B  A8  E9 0014D 18$:  BLBC    WDW_ZERO, 19$                               ;1700
                                        09              35  A8  E8 00151      BLBS    FIL_FOUND, 19$                              ;1702
                                        53                      69  D0 00155      MOVL    NAM_BLOCK+4, R3                         ;1704
                                        52              FF      A9  9A 00158      MOVZBL  NAM_BLOCK+3, R2
                                                        6A  16 0015C      JSB     SRH$PUT_OUTPUT
                                    34  A8              0101    8F  B0 0015E 19$:  MOVW    #257, FOUND_ANY                             ;1707
                                                            20  A8  D6 00164      INCL    FIL_TOTMAT                                  ;1708
                                                                1B  11 00167      BRB     21$                                         ;1609
                                        17              29      A8  E9 00169 20$:  BLBC    OUT_PRINTING, 21$                           ;1712
                                                            04  A8  D5 0016D      TSTL    WDW_SUB_CNTR
                                                                12  13 00170      BEQL    21$
                                        54              24      A8  D0 00172      MOVL    FIL_LINENUM, R4                             ;1715
                                        53                      6E  D0 00176      MOVL    REC_PTR, R3
                                        52              04      AE  D0 00179      MOVL    REC_SIZ, R2
                                                                6B  16 0017D      JSB     FORMAT_N_PUT
                                                                68  D4 0017F      CLRL    WDW_PRE_CNTR                                ;1716
                                                            04  A8  D7 00181      DECL    WDW_SUB_CNTR                                ;1717
```

```
                                 FE92  31 00184 21$:     BRW      1$                          ; 1609
                                       04 00187          RET                                 ; 1722

; Routine Size:  392 bytes,    Routine Base:  $CODE$ + 0AA2
```

```
;  1740     1723  1 ROUTINE std_match (size, buf) : JSB_STDMAT =
;  1741     1724  1
;  1742     1725  1 !++
;  1743     1726  1 ! Functional description
;  1744     1727  1 !
;  1745     1728  1 !       This routine checks to see if any of the search strings
;  1746     1729  1 !       is present in the current record
;  1747     1730  1 !
;  1748     1731  1 ! Calling sequence
;  1749     1732  1 !
;  1750     1733  1 !       std_match(.size, buf);
;  1751     1734  1 !
;  1752     1735  1 ! Input parameters
;  1753     1736  1 !
;  1754     1737  1 !       size = the length of the record
;  1755     1738  1 !       buf  = the address of the record
;  1756     1739  1 !
;  1757     1740  1 ! Implicit inputs
;  1758     1741  1 !
;  1759     1742  1 !       srhstr_desc     - The array of search string descriptors
;  1760     1743  1 !       cnt_srhstr      - The count of search string descriptors
;  1761     1744  1 !
;  1762     1745  1 ! Output parameters
;  1763     1746  1 !
;  1764     1747  1 !       none
;  1765     1748  1 !
;  1766     1749  1 ! Implicit outputs
;  1767     1750  1 !
;  1768     1751  1 !       none
;  1769     1752  1 !
;  1770     1753  1 ! Routine value
;  1771     1754  1 !
;  1772     1755  1 !       0 - None of the search strings was present
;  1773     1756  1 !       1 - At least one search string was present
;  1774     1757  1 !
;  1775     1758  1 ! Side effects
;  1776     1759  1 !
;  1777     1760  1 !       none
;  1778     1761  1 !
;  1779     1762  1 !--
;  1780     1763  1 (
;  1781     1764  2 BEGIN
;  1782     1765  2
;  1783     1766  2 LOCAL
;  1784     1767  2     rec_ptr     : REF BLOCK [,BYTE],            ! Pointer to string
;  1785     1768  2     upc_buf     : BLOCK [io_buff_sz, BYTE],     ! In case we must upcase
;  1786     1769  2     match_found;                                ! String found flag
;  1787     1770  2
;  1788     1771  2
;  1789     1772  2     IF .qua_exact
;  1790     1773  2     THEN
;  1791     1774  2         rec_ptr = .buf                          ! Get the input record address
;  1792     1775  2     ELSE
;  1793     1776  3         BEGIN
;  1794     1777  3         upcase(.size, .buf, upc_buf);           ! Upcase the record
;  1795     1778  3         rec_ptr = upc_buf;                      ! Point it to new version
;  1796     1779  2         END;
```

```
;  1797          1780  2
;  1798          1781  2      match_found = .mat_and;             ! 0 if OR or NOR, 1 if AND or NAND
;  1799          1782  2
;  1800          1783  2      INCR sdx FROM 0 TO .cnt_srhstr-1
;  1801          1784  2      DO
;  1802          1785  3          BEGIN
;  1803          1786  3          LOCAL
;  1804          1787  3              ptr;                                ! Pointer to found string
;  1805          1788  3
;  1806          1789  3          ptr = CH$FIND_SUB(.size, .rec_ptr,
;  1807          1790  3                            .srhstr_desc [.sdx, dsc$w_length],
;  1808          1791  3                            .srhstr_desc [.sdx, dsc$a_pointer]);
;  1809          1792  3
;  1810          1793  3          IF .mat_and             ! If AND or NAND
;  1811          1794  3          THEN
;  1812          1795  4                  BEGIN
;  1813          1796  4                  IF CH$FAIL(.ptr)
;  1814          1797  4                  THEN
;  1815          1798  5                      BEGIN
;  1816          1799  5                      match_found = false;
;  1817          1800  5                      EXITLOOP;
;  1818          1801  4                      END;
;  1819          1802  4                  END
;  1820          1803  3          ELSE                    ! Else OR or NOR
;  1821          1804  4                  BEGIN
;  1822          1805  4                  IF NOT CH$FAIL(.ptr)
;  1823          1806  4                  THEN
;  1824          1807  5                      BEGIN
;  1825          1808  5                      match_found = true;
;  1826          1809  5                      EXITLOOP;
;  1827          1810  4                      END;
;  1828          1811  3                  END;
;  1829          1812  2          END;
;  1830          1813  2
;  1831          1814  2      IF .mat_negate THEN match_found = NOT .match_found; ! Is it NOR or NAND?
;  1832          1815  2
;  1833        L 1816  2      %IF switch_statistics
;  1834          1817  2      %THEN
;  1835          1818  2          IF .match_found THEN stat_totmat = .stat_totmat+1; ! Bump match count
;  1836          1819  2      %FI
;  1837          1820  2
;  1838          1821  2      RETURN .match_found;
;  1839          1822  1      END;


                7E              57  7D 00000 STD_MATCH:
                                            MOVQ    R7, -(SP)                   ; 1723
                7E              55  7D 00003            MOVQ    R5, -(SP)       ;
                5E      F800    CE  9E 00006            MOVAB   -2048(SP), SP   ;
                05      0000'   CF  E9 0000B            BLBC    QUA_EXACT, 1$   ; 1772
                58              55  D0 00010            MOVL    BUF, REC_PTR    ; 1774
                                0C  11 00013            BRB     2$              ;
                53              6E  9E 00015 1$:        MOVAB   UPC_BUF, R3     ; 1777
                51              54  7D 00018            MOVQ    SIZE, R1        ;
```

```
                                0000V 30 0001B          BSBW    UPCASE                                          ;
                             58   6E  9E 0001E          MOVAB   UPC_BUF, REC_PTR                                ; 1778
                             57 0000' CF  9A 00021 2$:   MOVZBL  MAT_AND, MATCH_FOUND                            ; 1781
                             55       01  CE 00026          MNEGL   #1, SDX                                     ; 1789
                                      2F  11 00029          BRB     6$
                                0000'CF45  7F 0002B 3$:     PUSHAQ  SRHSTR_DESC[SDX]                            ; 1790
                             56       9E  3C 00030          MOVZWL  @(SP)+, R6
                                0000'CF45  7F 00033         PUSHAQ  SRHSTR_DESC+4[SDX]                          ; 1791
                             50       9E  D0 00038          MOVL    @(SP)+, R0
        68          54       60       56  39 0003B          MATCHC  R6, (R0), SIZE, (REC_PTR)
                                      03  13 00040          BEQL    4$
                             53       56  D0 00042          MOVL    R6, R3
                             53       56  C2 00045 4$:      SUBL2   R6, R3
                             06 0000' CF  E9 00048          BLBC    MAT_AND, 5$                                 ; 1796
                                      0B  12 0004D          BNEQ    6$
                                      57  D4 0004F          CLRL    MATCH_FOUND                                 ; 1799
                                      0D  11 00051          BRB     7$                                          ; 1798
                                      05  13 00053 5$:      BEQL    6$                                          ; 1805
                             57       01  D0 00055          MOVL    #1, MATCH_FOUND                             ; 1808
                                      06  11 00058          BRB     7$                                          ; 1807
                    CB       55 0000' CF  F2 0005A 6$:      AOBLSS  CNT_SRHSTR, SDX, 3$                         ; 1783
                             03 0000' CF  E9 00060 7$:      BLBC    MAT_NEGATE, 8$                              ; 1814
                             57       57  D2 00065          MCOML   MATCH_FOUND, MATCH_FOUND
                             04       57  E9 00068 8$:      BLBC    MATCH_FOUND, 9$                             ; 1818
                                0000' CF  D6 0006B          INCL    STAT_TOTMAT
                             50       57  D0 0006F 9$:      MOVL    MATCH_FOUND, R0                             ; 1821
                             5E     0800  CE  9E 00072       MOVAB   2048(SP), SP                                ; 1822
                                    01E0  8F  BA 00077       POPR    #^M<R5,R6,R7,R8>
                                          05  0007B         RSB                                                 ;

; Routine Size:  124 bytes,    Routine Base:  $CODE$ + 0C2A
```

```
;  1841          1823  1 ROUTINE file_error (     msg,                        ! The SEARCH message code
;  1842          1824  1                          rms_status,                 ! The main RMS error code
;  1843          1825  1                          fabb : REF BLOCK [,BYTE],   ! Address of FAB for file
;  1844          1826  1                          stv) =                      ! The RMS STV value
;  1845          1827  2 BEGIN
;  1846          1828  2
;  1847          1829  2 LOCAL
;  1848          1830  2     nam_blk : REF BLOCK [,BYTE];          ! Pointer to the name block
;  1849          1831  2
;  1850          1832  2 nam_blk = .fabb [fab$l_nam];             ! Get pointer to the name block
;  1851          1833  2
;  1852          1834  2 IF .nam_blk [nam$b_rsl] GTRU 0
;  1853          1835  2 THEN
;  1854          1836  3     BEGIN
;  1855          1837  3     tmp_desc [dsc$w_length]  = .nam_blk [nam$b_rsl];     ! Create file name desc
;  1856          1838  3     tmp_desc [dsc$a_pointer] = .nam_blk [nam$l_rsa];     ! ...
;  1857          1839  3     END
;  1858          1840  2 ELSE IF .nam_blk [nam$b_esl] GTRU 0
;  1859          1841  2 THEN
;  1860          1842  3     BEGIN
;  1861          1843  3     tmp_desc [dsc$w_length]  = .nam_blk [nam$b_esl];     ! Create file name desc
;  1862          1844  3     tmp_desc [dsc$a_pointer] = .nam_blk [nam$l_esa];     ! ...
;  1863          1845  3     END
;  1864          1846  2 ELSE
;  1865          1847  3     BEGIN
;  1866          1848  3     tmp_desc [dsc$w_length]  = .fabb [fab$b_fns];        ! Create file name desc
;  1867          1849  3     tmp_desc [dsc$a_pointer] = .fabb [fab$l_fna];        ! ...
;  1868          1850  2     END;
;  1869          1851  2
;  1870          1852  2 IF .rms_status EQL rms$_dme                 ! Does RMS have enough memory?
;  1871          1853  2 THEN                                        ! No, print actual error
;  1872          1854  3     BEGIN
;  1873          1855  3     IF .wdw_pre_recs GTR 0                  ! Do we have a window?
;  1874          1856  3     THEN
;  1875          1857  3         SIGNAL_STOP (.msg, 1, tmp_desc, .rms_status, .stv, srh$_wdw_maxprev)
;  1876          1858  3     ELSE
;  1877          1859  3         SIGNAL_STOP (.msg, 1, tmp_desc, .rms_status, .stv, srh$_insvirmem);
;  1878          1860  3     END
;  1879          1861  2 ELSE
;  1880          1862  2     SIGNAL (.msg, 1, tmp_desc, .rms_status, .stv);
;  1881          1863  2
;  1882          1864  2 RETURN (.msg OR %X'10000000');              ! Don't let $EXIT signal also.
;  1883          1865  2
;  1884          1866  1 END;


                                0004 00000 FILE_ERROR:
                                           .WORD   Save R2                                          ; 1823
                    52    0000'  CF  9E 00002    MOVAB   TMP_DESC, R2                             ;
                    51      0C   AC  D0 00007    MOVL    FABB, R1                                 ; 1832
                    50      28   A1  D0 0000B    MOVL    40(R1), NAM_BLK                          ;
                            03   A0  95 0000F    TSTB    3(NAM_BLK)                               ; 1834
                                 0B  13 00012    BEQL    1$                                       ;
                    62      03   A0  9B 00014    MOVZBW  3(NAM_BLK), TMP_DESC                     ; 1837
```

```
                04   A2        04   A0  D0 00018          MOVL    4(NAM_BLK), TMP_DESC+4          ; 1838
                                     19  11 0001D          BRB     3$                              ; 1834
                          0B    A0   95  0001F 1$:        TSTB    11(NAM_BLK)                     ; 1840
                                0B   13  00022          BEQL    2$                              ;
                62        0B    A0   9B  00024          MOVZBW  11(NAM_BLK), TMP_DESC           ; 1843
                04   A2   0C    A0   D0  00028          MOVL    12(NAM_BLK), TMP_DESC+4         ; 1844
                                     09  11 0002D          BRB     3$                              ; 1840
                62        34    A1   9B  0002F 2$:        MOVZBW  52(R1), TMP_DESC                ; 1848
                04   A2   2C    A1   D0  00033          MOVL    44(R1), TMP_DESC+4              ; 1849
           000184D4   8F        08   AC  D1  00038 3$:        CMPL    RMS_STATUS, #99540              ; 1852
                                     2A  12  00040          BNEQ    6$                              ;
                          0000' CF   D5  00042          TSTL    WDW_PRE_RECS                    ; 1855
                                     08  15  00046          BLEQ    4$                              ;
                     00000000G  8F   DD  00048          PUSHL   #SRH$_WDW_MAXPREV               ; 1857
                                     06  11  0004E          BRB     5$                              ;
                     00D712F2   8F   DD  00050 4$:        PUSHL   #14095090                       ; 1859
                           10   AC   DD  00056 5$:        PUSHL   STV                             ;
                           08   AC   DD  00059          PUSHL   RMS_STATUS                      ;
                                52   DD  0005C          PUSHL   R2                              ;
                                01   DD  0005E          PUSHL   #1                              ;
                           04   AC   DD  00060          PUSHL   MSG                             ;
                00000000G  00        06  FB  00063          CALLS   #6, LIB$STOP                    ;
                                     14  11  0006A          BRB     7$                              ; 1852
                           10   AC   DD  0006C 6$:        PUSHL   STV                             ; 1862
                           08   AC   DD  0006F          PUSHL   RMS_STATUS                      ;
                                52   DD  00072          PUSHL   R2                              ;
                                01   DD  00074          PUSHL   #1                              ;
                           04   AC   DD  00076          PUSHL   MSG                             ;
                00000000G  00        05  FB  00079          CALLS   #5, LIB$SIGNAL                  ;
           50        04   AC 10000000 8F  C9  00080 7$:        BISL3   #268435456, MSG, R0             ; 1864
                                     04  00089          RET                                     ; 1866

; Routine Size:  138 bytes,    Routine Base:  $CODE$ + 0CA6
```

```
;  1886          1867  1 ROUTINE upcase(in_siz,in_ptr,out_ptr) : NOVALUE JSB_UPCASE =
;  1887          1868  1 !++
;  1888          1869  1 ! Functional description
;  1889          1870  1 !
;  1890          1871  1 !       This routine converts a string to uppercase.  In testing it
;  1891          1872  1 !       appears to be faster to do this sort of loop than execute the
;  1892          1873  1 !       MOVTC instruction on the 11/780.
;  1893          1874  1 !
;  1894          1875  1 ! Input parameters
;  1895          1876  1 !
;  1896          1877  1 !       in_siz = size of input record to convert
;  1897          1878  1 !       in_ptr = address input of record to convert
;  1898          1879  1 !       out_ptr = address of output record buffer
;  1899          1880  1 !
;  1900          1881  1 ! Output parameters
;  1901          1882  1 !
;  1902          1883  1 !       Input record copied to output record buffer and all
;  1903          1884  1 !       lowercase alphabetic characters converted to uppercase.
;  1904          1885  1 !
;  1905          1886  1 !--
;  1906          1887  2 BEGIN
;  1907          1888  2 REGISTER
;  1908          1889  2     char: BYTE;                                ! Character to test
;  1909          1890  2
;  1910          1891  2 DECR count FROM .in_siz-1 TO 0 DO             ! Upcase the characters
;  1911          1892  3         BEGIN
;  1912          1893  3         char = CH$RCHAR_A(in_ptr);             !  Get next character
;  1913          1894  3         IF .char GEQU 'a'                      !  Lower case letter?
;  1914          1895  3          AND .char LEQU 'z'
;  1915          1896  3         THEN
;  1916          1897  3                 char = .char - %O'40';         ! Convert to upper
;  1917          1898  3         CH$WCHAR_A(.char,out_ptr);             ! Move character to buffer
;  1918          1899  2         END;
;  1919          1900  1 END;


                                        15  11 00000 UPCASE:  BRB     3$                          ; 1891
                              50        82  90 00002 1$:      MOVB    (IN_PTR)+, CHAR             ; 1893
                        61    8F        50  91 00005          CMPB    CHAR, #97                   ; 1894
                                        09  1F 00009          BLSSU   2$                          ;
                        7A    8F        50  91 0000B          CMPB    CHAR, #122                  ; 1895
                                        03  1A 0000F          BGTRU   2$                          ;
                              50        20  82 00011          SUBB2   #32, CHAR                   ; 1897
                              83        50  90 00014 2$:      MOVB    CHAR, (OUT_PTR)+            ; 1898
                              E8        51  F4 00017 3$:      SOBGEQ  COUNT, 1$                   ; 1891
                                        05  0001A            RSB                                 ; 1900

; Routine Size:  27 bytes,    Routine Base:  $CODE$ + 0D30
```

```
;  1921     1901  1 ROUTINE format_n_put(in_siz, in_ptr, line) : NOVALUE JSB_FORPUT =
;  1922     1902  1 !++
;  1923     1903  1 ! Functional description
;  1924     1904  1 !
;  1925     1905  1 !       The main reason to reformat is to prevent matches in .EXE and
;  1926     1906  1 !       .OBJ files from setting terminals in strange states.  The
;  1927     1907  1 !       /FORMAT=PASSALL qualifier is included to allow for exotic
;  1928     1908  1 !       character sets.
;  1929     1909  1 !
;  1930     1910  1 !       This routine copies the input buffer to the output file.  If
;  1931     1911  1 !       PASSALL is in effect, the input buffer is simply copied to output.
;  1932     1912  1 !       If another format is active, a routine will be called to do the
;  1933     1913  1 !       reformatting.
;  1934     1914  1 !
;  1935     1915  1 ! Input parameters
;  1936     1916  1 !
;  1937     1917  1 !       in_siz = size of input record to convert
;  1938     1918  1 !       in_ptr = address of input record to convert
;  1939     1919  1 !       line = line number to print if .NUMBERS is true
;  1940     1920  1 !
;  1941     1921  1 ! Outputs
;  1942     1922  1 !
;  1943     1923  1 !       The line will go to the output file if one exists.
;  1944     1924  1 !
;  1945     1925  1 !--
;  1946     1926  2 BEGIN
;  1947     1927  2
;  1948     1928  2 LOCAL
;  1949     1929  2     len,                                    ! Output length
;  1950     1930  2     op  : REF BLOCK [,BYTE];                ! Local output pointer
;  1951     1931  2
;  1952     1932  2 IF NOT .out_printing
;  1953     1933  2 THEN
;  1954     1934  2     RETURN;                                 ! Nothing at all to do
;  1955     1935  2
;  1956     1936  3 IF (.format EQL form_passall) AND (NOT .qua_numbers)
;  1957     1937  2 THEN
;  1958     1938  3     BEGIN
;  1959     1939  3     srh$put_output(.in_siz, .in_ptr);       ! Move the buffer.
;  1960     1940  3     RETURN;                                 ! And go back to caller
;  1961     1941  2     END;
;  1962     1942  2
;  1963     1943  2 !
;  1964     1944  2 ! We will have to copy the buffer for line numbers or reformatting, or both
;  1965     1945  2 !
;  1966     1946  2 len = .in_siz;                              ! Nothing output yet
;  1967     1947  2 op = .output_buff;                          ! Set to start of output
;  19[illegible]    1948  2
;  1[illegible]     1949  2 IF .qua_numbers
;  1970     1950  2 THEN
;  1971     1951  3     BEGIN
;  1972     1952  3     LOCAL
;  1973     1953  3         stat,
;  1974     1954  3         tmplen;
;  1975     1955  3     tmp_desc [dsc$w_length] = 7;
;  1976     1956  3     tmp_desc [dsc$a_pointer] = .op;
;  1977     1957  4     IF NOT (stat = $FAO(%ASCID '!6SL    ', 0, tmp_desc, .line))
```

```
;  1978          1958  3     THEN
;  1979          1959  3         SIGNAL_STOP (.stat);     ! Signal the error
;  1980          1960  3     len  = 7+.len;               ! Bump our local length.
;  1981          1961  3     op   = 7+.op;                ! Bump the pointer to the output
;  1982          1962  2     END;
;  1983          1963  2
;  1984          1964  2 CASE .format FROM form_text TO form_nonulls OF
;  1985          1965  2     SET
;  1986          1966  2     [form_text]:
;  1987          1967  2                 len = format_action_text (.in_siz, .in_ptr, .op);
;  1988          1968  2     [form_passall]:
;  1989          1969  2                 CH$MOVE(.in_siz, .in_ptr, .op); ! Move the input buffer
;  1990          1970  2     [form_dump, form_nonulls]:
;  1991          1971  2                 len = format_action_dump (.in_siz, .in_ptr, .op);
;  1992          1972  2     TES;
;  1993          1973  2
;  1994          1974  2 srh$put_output(.len, .output_buff);          ! and then print
;  1995          1975  2
;  1996          1976  2 RETURN;
;  1997          1977  2
;  1998          1978  1 END;


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                                00  00  00  09  4C  53  36  21  002A4 P.ACB:   .ASCII  \!6SL\<9><0><0><0>                ;
                                                    010E0005  002AC P.ACA:   .LONG   17694725                          ;
                                                    00000000' 002B0          .ADDRESS P.ACB                            ;


                                                        .PSECT  $CODE$,NOWRT,2

                                    03E0    8F   BB 00000 FORMAT_N_PUT:
                                                          PUSHR   #^M<R5,R6,R7,R8,R9>                                 ; 1901
                                57                  52   7D 00004          MOVQ    R2, R7                               ;
                                03              0000' CF   E8 00007          BLBS    OUT_PRINTING, 1$                     ; 1932
                                                0083  31   0000C          BRW     12$                                  ;
                                01              0000' CF   D1 0000F 1$:      CMPL    FORMAT, #1                           ; 1936
                                                    0A  12   00014          BNEQ    2$                                   ;
                                05              0000' CF   E8 00016          BLBS    QUA_NUMBERS, 2$                      ;
                                52                  57   7D 0001B          MOVQ    IN_SIZ, R2                           ; 1939
                                                    6F   11   0001E          BRB     11$                                  ;
                                59                  57   D0 00020 2$:      MOVL    IN_SIZ, LEN                          ; 1946
                                56              0000' CF   D0 00023          MOVL    OUTPUT_BUFF, OP                      ; 1947
                                2F              0000' CF   E9 00028          BLBC    QUA_NUMBERS, 4$                      ; 1949
                        0000'   CF                  07   B0 0002D          MOVW    #7, TMP_DESC                         ; 1955
                        0000'   CF                  56   D0 00032          MOVL    OP, TMP_DESC+4                       ; 1956
                                                    54   DD 00037          PUSHL   LINE                                 ; 1957
                                                0000' CF   9F 00039          PUSHAB  TMP_DESC                             ;
                                                    7E   D4 0003D          CLRL    -(SP)                                ;
                                                0000' CF   9F 0003F          PUSHAB  P.ACA                                ;
                    00000000G   00                  04   FB 00043          CALLS   #4, SYS$FAO                          ;
                                09                  50   E8 0004A          BLBS    STAT, 3$                             ;
                                                    50   DD 0004D          PUSHL   STAT                                 ; 1959
                    00000000G   00                  01   FB 0004F          CALLS   #1, LIB$STOP                         ;
```

```
                        59          07  C0 00056 3$:    ADDL2   #7, LEN                          ; 1960
                        56          07  C0 00059        ADDL2   #7, OP                           ; 1961
                    03  00    0000' CF  CF 0005C 4$:    CASEL   FORMAT, #0, #3                   ; 1964
          0019      0019      0013      0008     00062 5$:    .WORD   6$-5$,-                    ;
                                                                 7$-5$,-                          ;
                                                                 8$-5$,-                          ;
                                                                 8$-5$                            ;
                        52          56  D0 0006A 6$:    MOVL    OP, R2                           ; 1967
                        54          57  7D 0006D        MOVQ    IN_SIZ, R4                       ;
                              0000V 30  00070          BSBW    FORMAT_ACTION_TEXT               ;
                                    0F  11 00073          BRB     9$                           ;
                    66              68  57  28 00075 7$:    MOVC3   IN_SIZ, (IN_PTR), (OP)          ; 1969
                                        0C  11 00079          BRB     10$                          ;
                        50          56  D0 0007B 8$:    MOVL    OP, R0                           ; 1971
                        52          57  7D 0007E        MOVQ    IN_SIZ, R2                       ;
                              0000V 30  00081          BSBW    FORMAT_ACTION_DUMP               ;
                        59          50  D0 00084 9$:    MOVL    R0, LEN                          ;
                        53    0000' CF  D0 00087 10$:   MOVL    OUTPUT_BUFF, R3                  ; 1974
                        52          59  D0 0008C        MOVL    LEN, R2                          ;
                              0000V 30  0008F 11$:    BSBW    SRH$PUT_OUTPUT                   ;
                              03E0  8F  BA 00092 12$:   POPR    #^M<R5,R6,R7,R8,R9>              ; 1978
                                        05 00096        RSB                                      ;

; Routine Size:  151 bytes,    Routine Base:  $CODE$ + 0D4B
```

```
;  2000     1979  1 ROUTINE format_action_text (len, inp, outp) : JSB_FORACTTEX =
;  2001     1980  2 BEGIN
;  2002     1981  2
;  2003     1982  2 ! Define a table of substitute strings for control characters.  We define a byte vector of offsets to
;  2004     1983  2 ! ASCIC substitute strings.  A zero in the table means no substitution, an non-zero is the offset from
;  2005     1984  2 ! the base of the substitute strings.
;  2006     1985  2 !
;  2007     1986  2 PSECT
;  2008     1987  2     PLIT = $OWN$_CRF (WRITE);           ! Place string in crf along with table
;  2009     1988  2 BIND
;  2010     1989  2     table_base = UPLIT BYTE (0);
;  2011     1990  2 OWN_CRF
;  2012     1991  2     table : VECTOR [256, BYTE] PRESET (
;  2013     1992  2             [%x'00'] =  (UPLIT BYTE (%asCIC 'NUL') - table_base),
;  2014     1993  2             [%x'01'] =  (UPLIT BYTE (%asCIC 'SOH') - table_base),
;  2015     1994  2             [%x'02'] =  (UPLIT BYTE (%asCIC 'STX') - table_base),
;  2016     1995  2             [%x'03'] =  (UPLIT BYTE (%asCIC 'ETX') - table_base),
;  2017     1996  2             [%x'04'] =  (UPLIT BYTE (%asCIC 'EOT') - table_base),
;  2018     1997  2             [%x'05'] =  (UPLIT BYTE (%asCIC 'ENQ') - table_base),
;  2019     1998  2             [%x'06'] =  (UPLIT BYTE (%asCIC 'ACK') - table_base),
;  2020     1999  2             [%x'07'] =  (UPLIT BYTE (%asCIC 'BEL') - table_base),
;  2021     2000  2             [%x'08'] =  (UPLIT BYTE (%asCIC 'BS') - table_base),
;  2022     2001  2             [%x'0E'] =  (UPLIT BYTE (%asCIC 'SO') - table_base),
;  2023     2002  2             [%x'0F'] =  (UPLIT BYTE (%asCIC 'SI') - table_base),
;  2024     2003  2             [%x'10'] =  (UPLIT BYTE (%asCIC 'DLE') - table_base),
;  2025     2004  2             [%x'11'] =  (UPLIT BYTE (%asCIC 'DC1') - table_base),
;  2026     2005  2             [%x'12'] =  (UPLIT BYTE (%asCIC 'DC2') - table_base),
;  2027     2006  2             [%x'13'] =  (UPLIT BYTE (%asCIC 'DC3') - table_base),
;  2028     2007  2             [%x'14'] =  (UPLIT BYTE (%asCIC 'DC4') - table_base),
;  2029     2008  2             [%x'15'] =  (UPLIT BYTE (%asCIC 'NAK') - table_base),
;  2030     2009  2             [%x'16'] =  (UPLIT BYTE (%asCIC 'SYN') - table_base),
;  2031     2010  2             [%x'17'] =  (UPLIT BYTE (%asCIC 'ETB') - table_base),
;  2032     2011  2             [%x'18'] =  (UPLIT BYTE (%asCIC 'CAN') - table_base),
;  2033     2012  2             [%x'19'] =  (UPLIT BYTE (%asCIC 'EM') - table_base),
;  2034     2013  2             [%x'1A'] =  (UPLIT BYTE (%asCIC 'SUB') - table_base),
;  2035     2014  2             [%x'1B'] =  (UPLIT BYTE (%asCIC 'ESC') - table_base),
;  2036     2015  2             [%x'1C'] =  (UPLIT BYTE (%asCIC 'FS') - table_base),
;  2037     2016  2             [%x'1D'] =  (UPLIT BYTE (%asCIC 'GS') - table_base),
;  2038     2017  2             [%x'1E'] =  (UPLIT BYTE (%asCIC 'RS') - table_base),
;  2039     2018  2             [%x'1F'] =  (UPLIT BYTE (%asCIC 'US') - table_base),
;  2040     2019  2             [%x'7F'] =  (UPLIT BYTE (%asCIC 'DEL') - table_base),
;  2041     2020  2             [%x'80'] =  (UPLIT BYTE (%asCIC 'x80') - table_base),
;  2042     2021  2             [%x'81'] =  (UPLIT BYTE (%asCIC 'x81') - table_base),
;  2043     2022  2             [%x'82'] =  (UPLIT BYTE (%asCIC 'x82') - table_base),
;  2044     2023  2             [%x'83'] =  (UPLIT BYTE (%asCIC 'x83') - table_base),
;  2045     2024  2             [%x'84'] =  (UPLIT BYTE (%asCIC 'IND') - table_base),
;  2046     2025  2             [%x'85'] =  (UPLIT BYTE (%asCIC 'NEL') - table_base),
;  2047     2026  2             [%x'86'] =  (UPLIT BYTE (%asCIC 'SSA') - table_base),
;  2048     2027  2             [%x'C7'] =  (UPLIT BYTE (%asCIC 'ESA') - table_base),
;  2049     2028  2             [%x'88'] =  (UPLIT BYTE (%asCIC 'HTS') - table_base),
;  2050     2029  2             [%x'89'] =  (UPLIT BYTE (%asCIC 'HTJ') - table_base),
;  2051     2030  2             [%x'8A'] =  (UPLIT BYTE (%asCIC 'VTS') - table_base),
;  2052     2031  2             [%x'8B'] =  (UPLIT BYTE (%asCIC 'PLD') - table_base),
;  2053     2032  2             [%x'8C'] =  (UPLIT BYTE (%asCIC 'PLU') - table_base),
;  2054     2033  2             [%x'8D'] =  (UPLIT BYTE (%asCIC 'RI') - table_base),
;  2055     2034  2             [%x'8E'] =  (UPLIT BYTE (%asCIC 'SS2') - table_base),
;  2056     2035  2             [%x'8F'] =  (UPLIT BYTE (%asCIC 'SS3') - table_base),
```

```
;  2057          2036  2              [%x'90'] =  (UPLIT BYTE (%ascic 'DCS') - table_base),
;  2058          2037  2              [%x'91'] =  (UPLIT BYTE (%ascic 'PU1') - table_base),
;  2059          2038  2              [%x'92'] =  (UPLIT BYTE (%ascic 'PU2') - table_base),
;  2060          2039  2              [%x'93'] =  (UPLIT BYTE (%ascic 'STS') - table_base),
;  2061          2040  2              [%x'94'] =  (UPLIT BYTE (%ascic 'CCH') - table_base),
;  2062          2041  2              [%x'95'] =  (UPLIT BYTE (%ascic 'MW') - table_base),
;  2063          2042  2              [%x'96'] =  (UPLIT BYTE (%ascic 'SPA') - table_base),
;  2064          2043  2              [%x'97'] =  (UPLIT BYTE (%ascic 'EPA') - table_base),
;  2065          2044  2              [%x'98'] =  (UPLIT BYTE (%ascic 'x98') - table_base),
;  2066          2045  2              [%x'99'] =  (UPLIT BYTE (%ascic 'x99') - table_base),
;  2067          2046  2              [%x'9A'] =  (UPLIT BYTE (%ascic 'x9A') - table_base),
;  2068          2047  2              [%x'9B'] =  (UPLIT BYTE (%ascic 'CSI') - table_base),
;  2069          2048  2              [%x'9C'] =  (UPLIT BYTE (%ascic 'ST') - table_base),
;  2070          2049  2              [%x'9D'] =  (UPLIT BYTE (%ascic 'OSC') - table_base),
;  2071          2050  2              [%x'9E'] =  (UPLIT BYTE (%ascic 'PM') - table_base),
;  2072          2051  2              [%x'9F'] =  (UPLIT BYTE (%ascic 'APC') - table_base),
;  2073          2052  2              [%x'A0'] =  (UPLIT BYTE (%ascic 'xA0') - table_base),
;  2074          2053  2              [%x'FF'] =  (UPLIT BYTE (%ascic 'xFF') - table_base));
;  2075          2054  2  BIND
;  2076          2055  2      table_top = UPLIT BYTE (0);                 ! Hang a label on the end
;  2077          2056  2
;  2078          2057  2  ! Make sure that all of the strings total fewer than 256 bytes, so that byte offsets will work.  Note
;  2079          2058  2  ! that BLISS stores the above table like <table-base><ascic-strings><table><table-top> so that we must
;  2080          2059  2  ! include the length of the table itself.  Also test the assumption about storage format.  (We have
;  2081          2060  2  ! defined both OWN and PLIT to the same psect.)
;  2082          2061  2  !
;  2083          2062  2  $assume ((table_top-table_base) LEQ 511);
;  2084          2063  2  $assume ((table_base LSSA table) AND (table LSSA table_top));
;  2085          2064  2
;  2086          2065  2  REGISTER
;  2087          2066  2      ip,                                         ! Input pointer
;  2088          2067  2      op;                                         ! Output pointer
;  2089          2068  2
;  2090          2069  2  ip = .inp;
;  2091          2070  2  op = .outp;
;  2092          2071  2
;  2093          2072  2  DECR count FROM .len-1 TO 0                    ! Convert the controls
;  2094          2073  2  DO
;  2095          2074  3      BEGIN
;  2096          2075  3      REGISTER
;  2097          2076  3          char,                                   ! Local character variable
;  2098          2077  3          string : REF VECTOR [, BYTE];           ! Pointer to string for expansion
;  2099          2078  3
;  2100          2079  3      char = CH$RCHAR_A (ip);                     ! Get next character
;  2101          2080  3      IF (string = .table [.char]) NEQ 0          ! See if the substitution offset is zero
;  2102          2081  3      THEN
;  2103          2082  4          BEGIN
;  2104          2083  4          REGISTER
;  2105          2084  4              len;
;  2106          2085  4          string = .string + table_base;          ! Turn the offset into an address
;  2107          2086  4          CH$WCHAR_A ('<', op);                   ! Start with the open bracket
;  2108          2087  4          len = .string [0];                      ! Move the length to a register
;  2109          2088  4          CH$MOVE (.len, string [1], .op);        ! Copy the ASCIC string
;  2110          2089  4          op = .op+.len;                          ! Move the output pointer
;  2111          2090  4          CH$WCHAR_A ('>', op);                   ! And finish with the close bracket
;  2112          2091  4          END
;  2113          2092  3      ELSE                                        ! Offset is zero, just move the char
```

```
;  2114          2093  3          CH$WCHAR_A (.char, op);
;  2115          2094  2      END;
;  2116          2095  2
;  2117          2096  2 RETURN .op-.output_buff;                    ! Pass the new length back
;  2118          2097  1 END;


                                                .PSECT  $OWN$_CRF,NOEXE,2

                                        00  0032C P.ACC:   .BYTE   0
                                4C  55  4E  03  0032D P.ACD:   .ASCII  <3>\NUL\
                                48  4F  53  03  00331 P.ACE:   .ASCII  <3>\SOH\
                                58  54  53  03  00335 P.ACF:   .ASCII  <3>\STX\
                                58  54  45  03  00339 P.ACG:   .ASCII  <3>\ETX\
                                54  4F  45  03  0033D P.ACH:   .ASCII  <3>\EOT\
                                51  4E  45  03  00341 P.ACI:   .ASCII  <3>\ENQ\
                                4B  43  41  03  00345 P.ACJ:   .ASCII  <3>\ACK\
                                4C  45  42  03  00349 P.ACK:   .ASCII  <3>\BEL\
                                    53  42  02  0034D P.ACL:   .ASCII  <2>\BS\
                                    4F  53  02  00350 P.ACM:   .ASCII  <2>\SO\
                                    49  53  02  00353 P.ACN:   .ASCII  <2>\SI\
                                45  4C  44  03  00356 P.ACO:   .ASCII  <3>\DLE\
                                31  43  44  03  0035A P.ACP:   .ASCII  <3>\DC1\
                                32  43  44  03  0035E P.ACQ:   .ASCII  <3>\DC2\
                                33  43  44  03  00362 P.ACR:   .ASCII  <3>\DC3\
                                34  43  44  03  00366 P.ACS:   .ASCII  <3>\DC4\
                                4B  41  4E  03  0036A P.ACT:   .ASCII  <3>\NAK\
                                4E  59  53  03  0036E P.ACU:   .ASCII  <3>\SYN\
                                42  54  45  03  00372 P.ACV:   .ASCII  <3>\ETB\
                                4E  41  43  03  00376 P.ACW:   .ASCII  <3>\CAN\
                                    4D  45  02  0037A P.ACX:   .ASCII  <2>\EM\
                                42  55  53  03  0037D P.ACY:   .ASCII  <3>\SUB\
                                43  53  45  03  00381 P.ACZ:   .ASCII  <3>\ESC\
                                    53  46  02  00385 P.ADA:   .ASCII  <2>\FS\
                                    53  47  02  00388 P.ADB:   .ASCII  <2>\GS\
                                    53  52  02  0038B P.ADC:   .ASCII  <2>\RS\
                                    53  55  02  0038E P.ADD:   .ASCII  <2>\US\
                                4C  45  44  03  00391 P.ADE:   .ASCII  <3>\DEL\
                                30  38  78  03  00395 P.ADF:   .ASCII  <3>\x80\
                                31  38  78  03  00399 P.ADG:   .ASCII  <3>\x81\
                                32  38  78  03  0039D P.ADH:   .ASCII  <3>\x82\
                                33  38  78  03  003A1 P.ADI:   .ASCII  <3>\x83\
                                44  4E  49  03  003A5 P.ADJ:   .ASCII  <3>\IND\
                                4C  45  4E  03  003A9 P.ADK:   .ASCII  <3>\NEL\
                                41  53  53  03  003AD P.ADL:   .ASCII  <3>\SSA\
                                41  53  45  03  003B1 P.ADM:   .ASCII  <3>\ESA\
                                53  54  48  03  003B5 P.ADN:   .ASCII  <3>\HTS\
                                4A  54  48  03  003B9 P.ADO:   .ASCII  <3>\HTJ\
                                53  54  56  03  003BD P.ADP:   .ASCII  <3>\VTS\
                                44  4C  50  03  003C1 P.ADQ:   .ASCII  <3>\PLD\
                                55  4C  50  03  003C5 P.ADR:   .ASCII  <3>\PLU\
                                    49  52  02  003C9 P.ADS:   .ASCII  <2>\RI\
                                32  53  53  03  003CC P.ADT:   .ASCII  <3>\SS2\
                                33  53  53  03  003D0 P.ADU:   .ASCII  <3>\SS3\
                                53  43  44  03  003D4 P.ADV:   .ASCII  <3>\DCS\
                                31  55  50  03  003D8 P.ADW:   .ASCII  <3>\PU1\
```

```
                                                32  55  50  03  003DC P.ADX:   .ASCII  <3>\PU2\                                ;
                                                53  54  53  03  003E0 P.ADY:   .ASCII  <3>\STS\                                ;
                                                48  43  43  03  003E4 P.ADZ:   .ASCII  <3>\CCH\                                ;
                                                    57  4D  02  003E8 P.AEA:   .ASCII  <2>\MW\                                 ;
                                                41  50  53  03  003EB P.AEB:   .ASCII  <3>\SPA\                                ;
                                                41  50  45  03  003EF P.AEC:   .ASCII  <3>\EPA\                                ;
                                                38  39  78  03  003F3 P.AED:   .ASCII  <3>\x98\                                ;
                                                39  39  78  03  003F7 P.AFE:   .ASCII  <3>\x99\                                ;
                                                41  39  78  03  003FB P.AEF:   .ASCII  <3>\x9A\                                ;
                                                49  53  43  03  003FF P.AEG:   .ASCII  <3>\CSI\                                ;
                                                    54  53  02  00403 P.AEH:   .ASCII  <2>\ST\                                 ;
                                                43  53  4F  03  00406 P.AEI:   .ASCII  <3>\OSC\                                ;
                                                    4D  50  02  0040A P.AEJ:   .ASCII  <2>\PM\                                 ;
                                                43  50  41  03  0040D P.AEK:   .ASCII  <3>\APC\                                ;
                                                30  41  78  03  00411 P.AEL:   .ASCII  <3>\xA0\                                ;
                                                46  46  78  03  00415 P.AEM:   .ASCII  <3>\xFF\                                ;
                                                            00419               .BLKB   3                                       ;
                                        21  1D  19  15  11  0D  09  05  01  0041C TABLE:    .BYTE   1, 5, 9, 13, 17, 21, 25, 29, 33         ;
                                                            00# 00425           .BYTE   0[5]                                    ;
59  55  51  4E  4A  46  42  3E  3A  36  32  2E  2A  27  24  0042A               .BYTE   36, 39, 42, 46, 50, 54, 58, 62, 66, 70, -   ;
                                                62  5F  5C  00439                       74, 78, 81, 85, 89, 92, 95, 98                  ;
                                                            00# 0043C           .BYTE   0[95]                                   ;
9D  99  95  91  8D  89  85  81  7D  79  75  71  6D  69  65  0049B               .BYTE   101, 105, 109, 113, 117, 121, 125, -127, -  ;
D7  D3  CF  CB  C7  C3  BF  BC  B8  B4  B0  AC  A8  A4  A0  004AA                       -123, -119, -115, -111, -107, -103, -99, -      ;
                                            E5  E1  DE  DA  004B9                       -96, -92, -88, -84, -80, -76, -72, -68, -       ;
                                                                                        -65, -61, -57, -53, -49, -45, -41, -38, -       ;
                                                                                        -34, -31, -27                                   ;
                                                            00# 004BD           .BYTE   0[94]                                   ;
                                                            E9  005:B           .BYTE   -23                                     ;
                                                            00  0051C P.AEN:    .BYTE   0                                       ;

                                                        TABLE_BASE=         P.ACC
                                                        TABLE_TOP=          P.AEN


                                                                    .PSECT  $CODE$,NOWRT,2

                    7E              58  7D 00000 FORMAT_ACTION_TEXT:
                                                    MOVQ    R8, -(SP)                   ; 1979
                    7E              56  7D 00003            MOVQ    R6, -(SP)           ;
                    56              55  D0 00006            MOVL    INP, IP             ; 2069
                    57              52  D0 00009            MOVL    OUTP, OP            ; 2070
                    59              54  D0 0000C            MOVL    LEN, COUNT          ; 2072
                                    29  11 0000F            BRB     4$                  ;
                    50              86  9A 00011 1$:        MOVZBL  (IP)+, CHAR         ; 2079
                    51      0000'CF40   9A 00014            MOVZBL  TABLE[CHAR], STRING ; 2080
                                    19  13 0001A            BEQL    2$                  ;
                    51      0000'CF41   9E 0001C            MOVAB   TABLE_BASE[STRING], STRING ; 2085
                    87              3C  90 00022            MOVB    #60, (OP)+          ; 2086
                    58              61  9A 00025            MOVZBL  (STRING), LEN       ; 2087
67          01      A1              58  28 00028            MOVC3   LEN, 1(STRING), (OP) ; 2088
                    57              58  C0 0002D            ADDL2   LEN, OP             ; 2089
                    67              3E  90 00030            MOVB    #62, (OP)           ; 2090
                                    03  11 00033            BRB     3$                  ; 2093
                    67              50  90 00035 2$:        MOVB    CHAR, (OP)          ;
                                    57  D6 00038 3$:        INCL    OP                  ; 2090
                    D4              59  F4 0003A 4$:        SOBGEQ  COUNT, 1$           ; 2072
```

```
                57      0000'  CF  C2 0003D          SUBL2   OUTPUT_BUFF, R7                    ; 2096
                50             57  D0 00042          MOVL    R7, R0                            ;
                       03C0    8F  BA 00045          POPR    #^M<R6,R7,R8,R9>                  ; 2097
                               05  00049          RSB                                       ;

; Routine Size:  74 bytes,    Routine Base:  $CODE$ + 0DE2
```

```
;  2120          2098  1 ROUTINE format_action_dump (len, inp, outp) : JSB_FORACTDUM =
;  2121          2099  2 BEGIN
;  2122          2100  2
;  2123          2101  2 ! Define a table of substitute strings for control characters.  We define a byte vector of offsets to
;  2124          2102  2 ! ASCIC substitute strings.  A zero in the table means no substitution, an non-zero is the offset from
;  2125          2103  2 ! the base of the substitute strings.
;  2126          2104  2 !
;  2127          2105  2 PSECT
;  2128          2106  2     PLIT = $OWN$_CRF (WRITE);           ! Place string in crf along with table
;  2129          2107  2 BIND
;  2130          2108  2     table_base = UPLIT BYTE (0);
;  2131          2109  2 OWN_CRF
;  2132          2110  2     table : VECTOR [256, BYTE] PRESET (
;  2133          2111  2             [%x'00'] =  (UPLIT BYTE (%ascic 'NUL') - table_base),
;  2134          2112  2             [%x'01'] =  (UPLIT BYTE (%ascic 'SOH') - table_base),
;  2135          2113  2             [%x'02'] =  (UPLIT BYTE (%ascic 'STX') - table_base),
;  2136          2114  2             [%x'03'] =  (UPLIT BYTE (%ascic 'ETX') - table_base),
;  2137          2115  2             [%x'04'] =  (UPLIT BYTE (%ascic 'EOT') - table_base),
;  2138          2116  2             [%x'05'] =  (UPLIT BYTE (%ascic 'ENQ') - table_base),
;  2139          2117  2             [%x'06'] =  (UPLIT BYTE (%ascic 'ACK') - table_base),
;  2140          2118  2             [%x'07'] =  (UPLIT BYTE (%ascic 'BEL') - table_base),
;  2141          2119  2             [%x'08'] =  (UPLIT BYTE (%ascic 'BS') - table_base),
;  2142          2120  2             [%x'09'] =  (UPLIT BYTE (%ascic 'HT') - table_base),
;  2143          2121  2             [%x'0A'] =  (UPLIT BYTE (%ascic 'CR') - table_base),
;  2144          2122  2             [%x'0B'] =  (UPLIT BYTE (%ascic 'LF') - table_base),
;  2145          2123  2             [%x'0C'] =  (UPLIT BYTE (%ascic 'VT') - table_base),
;  2146          2124  2             [%x'0D'] =  (UPLIT BYTE (%ascic 'FF') - table_base),
;  2147          2125  2             [%x'0E'] =  (UPLIT BYTE (%ascic 'SO') - table_base),
;  2148          2126  2             [%x'0F'] =  (UPLIT BYTE (%ascic 'SI') - table_base),
;  2149          2127  2             [%x'10'] =  (UPLIT BYTE (%ascic 'DLE') - table_base),
;  2150          2128  2             [%x'11'] =  (UPLIT BYTE (%ascic 'DC1') - table_base),
;  2151          2129  2             [%x'12'] =  (UPLIT BYTE (%ascic 'DC2') - table_base),
;  2152          2130  2             [%x'13'] =  (UPLIT BYTE (%ascic 'DC3') - table_base),
;  2153          2131  2             [%x'14'] =  (UPLIT BYTE (%ascic 'DC4') - table_base),
;  2154          2132  2             [%x'15'] =  (UPLIT BYTE (%ascic 'NAK') - table_base),
;  2155          2133  2             [%x'16'] =  (UPLIT BYTE (%ascic 'SYN') - table_base),
;  2156          2134  2             [%x'17'] =  (UPLIT BYTE (%ascic 'ETB') - table_base),
;  2157          2135  2             [%x'18'] =  (UPLIT BYTE (%ascic 'CAN') - table_base),
;  2158          2136  2             [%x'19'] =  (UPLIT BYTE (%ascic 'EM') - table_base),
;  2159          2137  2             [%x'1A'] =  (UPLIT BYTE (%ascic 'SUB') - table_base),
;  2160          2138  2             [%x'1B'] =  (UPLIT BYTE (%ascic 'ESC') - table_base),
;  2161          2139  2             [%x'1C'] =  (UPLIT BYTE (%ascic 'FS') - table_base),
;  2162          2140  2             [%x'1D'] =  (UPLIT BYTE (%ascic 'GS') - table_base),
;  2163          2141  2             [%x'1E'] =  (UPLIT BYTE (%ascic 'RS') - table_base),
;  2164          2142  2             [%x'1F'] =  (UPLIT BYTE (%ascic 'US') - table_base),
;  2165          2143  2             [%x'7F'] =  (UPLIT BYTE (%ascic 'DEL') - table_base),
;  2166          2144  2             [%x'80'] =  (UPLIT BYTE (%ascic 'x80') - table_base),
;  2167          2145  2             [%x'81'] =  (UPLIT BYTE (%ascic 'x81') - table_base),
;  2168          2146  2             [%x'82'] =  (UPLIT BYTE (%ascic 'x82') - table_base),
;  2169          2147  2             [%x'83'] =  (UPLIT BYTE (%ascic 'x83') - table_base),
;  2170          2148  2             [%x'84'] =  (UPLIT BYTE (%ascic 'IND') - table_base),
;  2171          2149  2             [%x'85'] =  (UPLIT BYTE (%ascic 'NEL') - table_base),
;  2172          2150  2             [%x'86'] =  (UPLIT BYTE (%ascic 'SSA') - table_base),
;  2173          2151  2             [%x'87'] =  (UPLIT BYTE (%ascic 'ESA') - table_base),
;  2174          2152  2             [%x'88'] =  (UPLIT BYTE (%ascic 'HTS') - table_base),
;  2175          2153  2             [%x'89'] =  (UPLIT BYTE (%ascic 'HTJ') - table_base),
;  2176          2154  2             [%x'8A'] =  (UPLIT BYTE (%ascic 'VTS') - table_base),
```

```
;  2177          2155  2              [%x'8B'] =  (UPLIT BYTE (%ascic 'PLD') - table_base),
;  2178          2156  2              [%x'8C'] =  (UPLIT BYTE (%ascic 'PLU') - table_base),
;  2179          2157  2              [%x'8D'] =  (UPLIT BYTE (%ascic 'RI') - table_base),
;  2180          2158  2              [%x'8E'] =  (UPLIT BYTE (%ascic 'SS2') - table_base),
;  2181          2159  2              [%x'8F'] =  (UPLIT BYTE (%ascic 'SS3') - table_base),
;  2182          2160  2              [%x'90'] =  (UPLIT BYTE (%ascic 'DCS') - table_base),
;  2183          2161  2              [%x'91'] =  (UPLIT BYTE (%ascic 'PU1') - table_base),
;  2184          2162  2              [%x'92'] =  (UPLIT BYTE (%ascic 'PU2') - table_base),
;  2185          2163  2              [%x'93'] =  (UPLIT BYTE (%ascic 'STS') - table_base),
;  2186          2164  2              [%x'94'] =  (UPLIT BYTE (%ascic 'CCH') - table_base),
;  2187          2165  2              [%x'95'] =  (UPLIT BYTE (%ascic 'MW') - table_base),
;  2188          2166  2              [%x'96'] =  (UPLIT BYTE (%ascic 'SPA') - table_base),
;  2189          2167  2              [%x'97'] =  (UPLIT BYTE (%ascic 'EPA') - table_base),
;  2190          2168  2              [%x'98'] =  (UPLIT BYTE (%ascic 'x98') - table_base),
;  2191          2169  2              [%x'99'] =  (UPLIT BYTE (%ascic 'x99') - table_base),
;  2192          2170  2              [%x'9A'] =  (UPLIT BYTE (%asc'c 'x9A') - table_base),
;  2193          2171  2              [%x'9B'] =  (UPLIT BYTE (%ascic 'CSI') - table_base),
;  2194          2172  2              [%x'9C'] =  (UPLIT BYTE (%ascic 'ST') - table_base),
;  2195          2173  2              [%x'9D'] =  (UPLIT BYTE (%ascic 'OSC') - table_base),
;  2196          2174  2              [%x'9E'] =  (UPLIT BYTE (%ascic 'PM') - table_base),
;  2197          2175  2              [%x'9F'] =  (UPLIT BYTE (%ascic 'APC') - table_base),
;  2198          2176  2              [%x'A0'] =  (UPLIT BYTE (%ascic 'xA0') - table_base),
;  2199          2177  2              [%x'FF'] =  (UPLIT BYTE (%ascic 'xFF') - table_base));
;  2200          2178  2 BIND
;  2201          2179  2     table_top = UPLIT BYTE (0);                    ! Hang a label on the end
;  2202          2180  2
;  2203          2181  2 ! Make sure that all of the strings total fewer than 256 bytes, so that byte offsets will work.  Note
;  2204          2182  2 ! that BLISS stores the above table like <table-base><ascic-strings><table><table-top> so that we must
;  2205          2183  2 ! include the length of the table itself.  Also test the assumption about storage format.  (We have
;  2206          2184  2 ! defined both OWN and PLIT to the same psect.)
;  2207          2185  2 !
;  2208          2186  2 $assume ((table_top-table_base) LEQ 511);
;  2209          2187  2 $assume ((table_base LSSA table) AND (table LSSA table_top));
;  2210          2188  2
;  2211          2189  2 REGISTER
;  2212          2190  2     ip,                                            ! Input pointer
;  2213          2191  2     op;                                            ! Output pointer
;  2214          2192  2
;  2215          2193  2 ip = .inp;
;  2216          2194  2 op = .outp;
;  2217          2195  2
;  2218          2196  2 DECR count FROM .len-1 TO 0                       ! Convert the controls
;  2219          2197  2 DO
;  2220          2198  3     BEGIN
;  2221          2199  3     REGISTER
;  2222          2200  3         char,                                      ! Local character variable
;  2223          2201  3         string : REF VECTOR [, BYTE];              ! Pointer to string for expansion
;  2224          2202  3
;  2225          2203  3     char = CH$RCHAR_A (ip);                        ! Get next character
;  2226          2204  5     IF NOT ((.char EQL 0)                          ! Skip if character is zero
;  2227          2205  4             AND (.format EQL form_nonulls))        !  and we are skipping nulls
;  2228          2206  3     THEN
;  2229          2207  4         BEGIN
;  2230          2208  4         IF (string = .table [.char]) NEQ 0         ! See if the substitution offset is zero
;  2231          2209  4         THEN
;  2232          2210  5             BEGIN
;  2233          2211  5             REGISTER
```

```
;  2234     2212  5                 len;
;  2235     2213  5             string = .string + table_base;      ! Turn the offset into an address
;  2236     2214  5             CH$WCHAR_A ('<', op);               ! Start with the open bracket
;  2237     2215  5             len = .string [0];                  ! Move the length to a register
;  2238     2216  5             CH$MOVE (.len, string [1], .op);    ! Copy the ASCIC string
;  2239     2217  5             op = .op+.len;                      ! Move the output pointer
;  2240     2218  5             CH$WCHAR_A ('>', op);               ! And finish with the close bracket
;  2241     2219  5             END
;  2242     2220  4         ELSE                                    ! Offset is zero, just move the char
;  2243     2221  4             CH$WCHAR_A (.char, op);
;  2244     2222  3             END;
;  2245     2223  2     END;
;  2246     2224  2
;  2247     2225  2 RETURN .op-.output_buff;                       ! Pass the new length back
;  2248     2226  1 END;


                                                        .PSECT  $OWN$_CRF,NOEXE,2

                                                00  0051D P.AEO:   .BYTE   0                                       ;
                                            4C  55  4E  03  0051E P.AEP:   .ASCII  <3>\NUL\                         ;
                                            48  4F  53  03  00522 P.AEQ:   .ASCII  <3>\SOH\                         ;
                                            58  54  53  03  00526 P.AER:   .ASCII  <3>\STX\                         ;
                                            58  54  45  03  0052A P.AES:   .ASCII  <3>\ETX\                         ;
                                            54  4F  45  03  0052E P.AET:   .ASCII  <3>\EOT\                         ;
                                            51  4E  45  03  00532 P.AEU:   .ASCII  <3>\ENQ\                         ;
                                            4B  43  41  03  00536 P.AEV:   .ASCII  <3>\ACK\                         ;
                                            4C  45  42  03  0053A P.AEW:   .ASCII  <3>\BEL\                         ;
                                                53  42  02  0053E P.AEX:   .ASCII  <2>\BS\                          ;
                                                54  48  02  00541 P.AEY:   .ASCII  <2>\HT\                          ;
                                                52  43  02  00544 P.AEZ:   .ASCII  <2>\CR\                          ;
                                                46  4C  02  00547 P.AFA:   .ASCII  <2>\LF\                          ;
                                                54  56  02  0054A P.AFB:   .ASCII  <2>\VT\                          ;
                                                46  46  02  0054D P.AFC:   .ASCII  <2>\FF\                          ;
                                                4F  53  02  00550 P.AFD:   .ASCII  <2>\SO\                          ;
                                                49  53  02  00553 P.AFE:   .ASCII  <2>\SI\                          ;
                                            45  4C  44  03  00556 P.AFF:   .ASCII  <3>\DLE\                         ;
                                            31  43  44  03  0055A P.AFG:   .ASCII  <3>\DC1\                         ;
                                            32  43  44  03  0055E P.AFH:   .ASCII  <3>\DC2\                         ;
                                            33  43  44  03  00562 P.AFI:   .ASCII  <3>\DC3\                         ;
                                            34  43  44  03  00566 P.AFJ:   .ASCII  <3>\DC4\                         ;
                                            4B  41  4E  03  0056A P.AFK:   .ASCII  <3>\NAK\                         ;
                                            4E  59  53  03  0056E P.AFL:   .ASCII  <3>\SYN\                         ;
                                            42  54  45  03  00572 P.AFM:   .ASCII  <3>\ETB\                         ;
                                            4E  41  43  03  00576 P.AFN:   .ASCII  <3>\CAN\                         ;
                                                4D  45  02  0057A P.AFO:   .ASCII  <2>\EM\                          ;
                                            42  55  53  03  0057D P.AFP:   .ASCII  <3>\SUB\                         ;
                                            43  53  45  03  00581 P.AFQ:   .ASCII  <3>\ESC\                         ;
                                                53  46  02  00585 P.AFR:   .ASCII  <2>\FS\                          ;
                                                53  47  02  00588 P.AFS:   .ASCII  <2>\GS\                          ;
                                                53  52  02  0058B P.AFT:   .ASCII  <2>\RS\                          ;
                                                53  55  02  0058E P.AFU:   .ASCII  <2>\US\                          ;
                                            4C  45  44  03  00591 P.AFV:   .ASCII  <3>\DEL\                         ;
                                            30  38  78  03  00595 P.AFW:   .ASCII  <3>\x80\                         ;
                                            31  38  78  03  00599 P.AFX:   .ASCII  <3>\x81\                         ;
                                            32  38  78  03  0059D P.AFY:   .ASCII  <3>\x82\                         ;
```

```
                                33  38  78  03  005A1 P.AFZ:   .ASCII   <3>\x83\
                                44  4E  49  03  005A5 P.AGA:   .ASCII   <3>\IND\
                                4C  45  4E  03  005A9 P.AGB:   .ASCII   <3>\NEL\
                                41  53  53  03  005AD P.AGC:   .ASCII   <3>\SSA\
                                41  53  45  03  005B1 P.AGD:   .ASCII   <3>\ESA\
                                53  54  48  03  005B5 P.AGE:   .ASCII   <3>\HTS\
                                4A  54  48  03  005B9 P.AGF:   .ASCII   <3>\HTJ\
                                53  54  56  03  005BD P.AGG:   .ASCII   <3>\VTS\
                                44  4C  50  03  005C1 P.AGH:   .ASCII   <3>\PLD\
                                55  4C  50  03  005C5 P.AGI:   .ASCII   <3>\PLU\
                                    49  52  02  005C9 P.AGJ:   .ASCII   <2>\RI\
                                32  53  53  03  005CC P.AGK:   .ASCII   <3>\SS2\
                                33  53  53  03  005D0 P.AGL:   .ASCII   <3>\SS3\
                                53  43  44  03  005D4 P.AGM:   .ASCII   <3>\DCS\
                                31  55  50  03  005D8 P.AGN:   .ASCII   <3>\PU1\
                                32  55  50  03  005DC P.AGO:   .ASCII   <3>\PU2\
                                53  54  53  03  005E0 P.AGP:   .ASCII   <3>\STS\
                                48  43  43  03  005E4 P.AGQ:   .ASCII   <3>\CCH\
                                    57  4D  02  005E8 P.AGR:   .ASCII   <2>\MW\
                                41  50  53  03  005EB P.AGS:   .ASCII   <3>\SPA\
                                41  50  45  03  005EF P.AGT:   .ASCII   <3>\EPA\
                                38  39  78  03  005F3 P.AGU:   .ASCII   <3>\x98\
                                39  39  78  03  005F7 P.AGV:   .ASCII   <3>\x99\
                                41  39  78  03  005FB P.AGW:   .ASCII   <3>\x9A\
                                49  53  43  03  005FF P.AGX:   .ASCII   <3>\CSI\
                                    54  53  02  00603 P.AGY:   .ASCII   <2>\ST\
                                43  53  4F  03  00606 P.AGZ:   .ASCII   <3>\OSC\
                                    4D  50  02  0060A P.AHA:   .ASCII   <2>\PM\
                                43  50  41  03  0060D P.AHB:   .ASCII   <3>\APC\
                                30  41  78  03  00611 P.AHC:   .ASCII   <3>\xA0\
                                46  46  78  03  00615 P.AHD:   .ASCII   <3>\xFF\
                                                00619         .BLKB    3
33  30  2D  2A  27  24  21  1D  19  15  11  0D  09  05  01  0061C TABLE.   .BYTE    1, 5, 9, 13, 17, 21, 25, 29, 33, 36, 39, -
6B  68  64  60  5D  59  55  51  4D  49  45  41  3D  39  36  0062B                     42, 45, 48, 51, 54, 57, 61, 65, 69, 73, -
                                    71  6E  0063A                     77, 81, 85, 89, 93, 96, 100, 104, 107, -
                                                                      110, 113
                                        00# 0063C         .BYTE    0[95]
AC  A8  A4  A0  9C  98  94  90  8C  88  84  80  7C  78  74  0069B         .BYTE    116, 120, 124, -128, -124, -120, -116, -
E6  E2  DE  DA  D6  D2  CE  CB  C7  C3  BF  BB  B7  B3  AF  006AA                     -112, -108, -104, -100, -96, -92, -88, -
                                    F4  F0  ED  E9  006B9                     -84, -81, -77, -73, -69, -65, -61, -57, -
                                                                      -53, -50, -46, -42, -38, -34, -30, -26, -
                                                                      -23, -19, -16, -12
                                        00# 006BD         .BYTE    0[94]
                                        F8  0071B         .BYTE    -8
                                        00  0071C P.AHE:   .BYTE    0

                                                TABLE_BASE=         P.AEO
                                                TABLE_TOP=          P.AHE


                                                        .PSECT  $CODE$,NOWRT,2

                        7E              58  7D 00000 FORMAT_ACTION_DUMP:
                                                MOVQ    R8, -(SP)                      ; 2098
                        7E              56  7D 00003         MOVQ    R6, -(SP)
                        56              53  D0 00006         MOVL    INP, IP                ; 2193
                        57              50  D0 00009         MOVL    OUTP, OP               ; 2194
```

```
                                59              52  D0 0000C          MOVL    LEN, COUNT                          ; 2196
                                                32  11 0000F          BRB     5$                                  ;
                                50              86  9A 00011 1$:     MOVZBL  (IP)+, CHAR                         ; 2203
                                                07  12 00014        BNEQ    2$                                  ; 2204
                                03      0000'   CF  D1 00016        CMPL    FORMAT, #3                          ; 2205
                                                26  13 0001B        BEQL    5$                                  ;
                                51      0000'CF40  9A 0001D 2$:     MOVZBL  TABLE[CHAR], STRING                 ; 2208
                                                19  13 00023        BEQL    3$                                  ;
                                51      0000'CF41  9E 00025        MOVAB   TABLE_BASE[STRING], STRING          ; 2213
                                87              3C  90 0002B        MOVB    #60, (OP)+                          ; 2214
                                58              61  9A 0002E        MOVZBL  (STRING), LEN                       ; 2215
                        67      01      A1      58  28 00031        MOVC3   LEN, 1(STRING), (OP)                ; 2216
                                57              58  C0 00036        ADDL2   LEN, OP                             ; 2217
                                67              3E  90 00039        MOVB    #62, (OP)                           ; 2218
                                                03  11 0003C        BRB     4$                                  ; 2221
                                67              50  90 0003E 3$:     MOVB    CHAR, (OP)                          ;
                                                57  D6 00041 4$:     INCL    OP                                  ; 2218
                                CB              59  F4 00043 5$:     SOBGEQ  COUNT, 1$                           ; 2196
                                57      0000'   CF  C2 00046        SUBL2   OUTPUT_BUFF, R7                     ; 2225
                                50              57  D0 0004B        MOVL    R7, R0                              ;
                                        03C0    8F  BA 0004E        POPR    #^M<R6,R7,R8,R9>                    ; 2226
                                                    05 00052        RSB                                         ;

; Routine Size:  83 bytes,    Routine Base:  $CODE$ + 0E2C
```

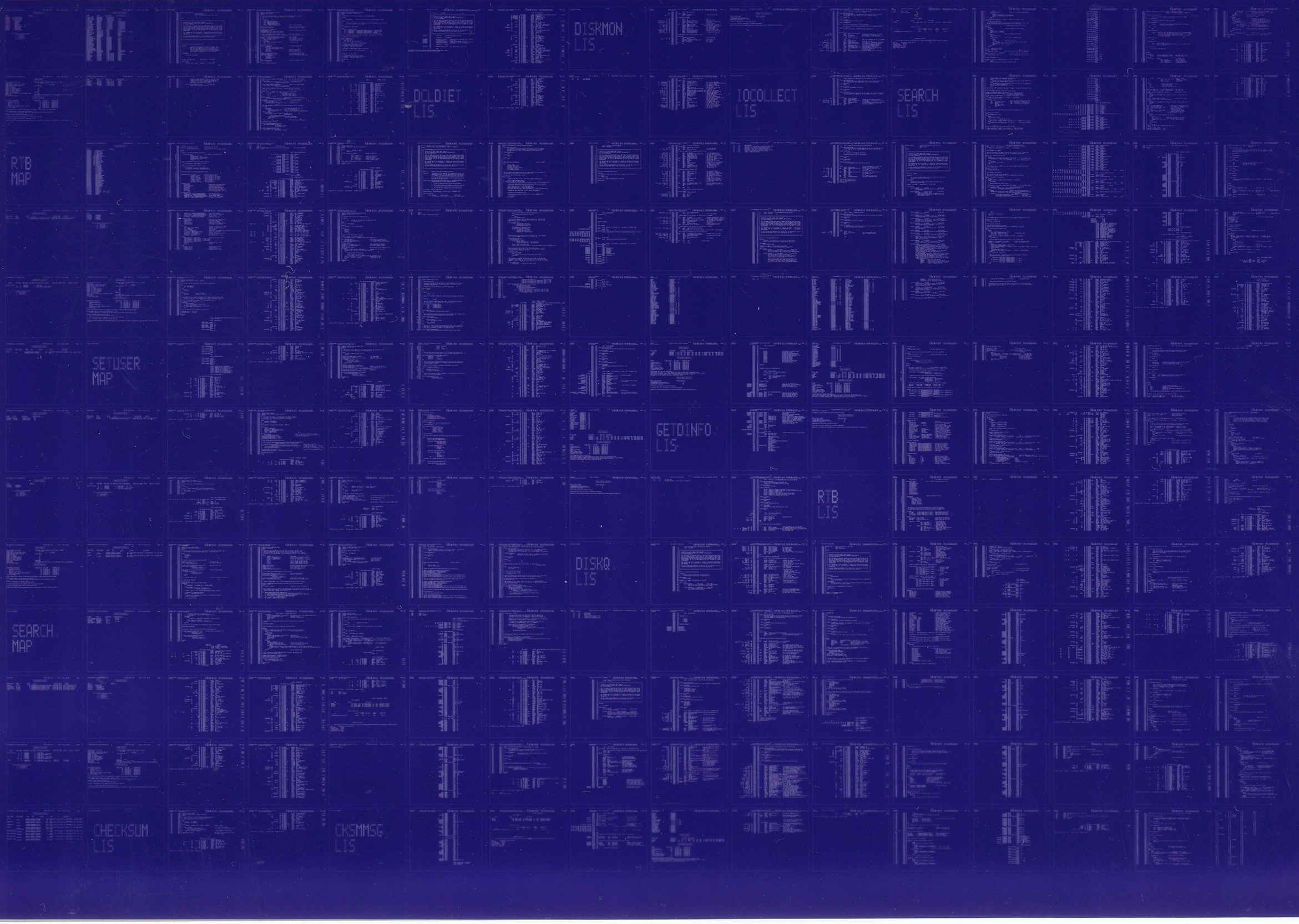
```
;  2250          2227  1 ROUTINE srh$put_output(length, buffer) : NOVALUE JSB_PUTOUT =
;  2251          2228  1 !++
;  2252          2229  1 ! Functional description
;  2253          2230  1 !
;  2254          2231  1 !       Routine to write records to the output file
;  2255          2232  1 !
;  2256          2233  1 ! Input parameters
;  2257          2234  1 !
;  2258          2235  1 !       length = length of string to output
;  2259          2236  1 !       buffer = address of string to output
;  2260          2237  1 !
;  2261          2238  1 !--
;  2262          2239  2 BEGIN
;  2263          2240  2
;  2264          2241  2 ROUTINE put (len, buf) : NOVALUE =
;  2265          2242  3 BEGIN
;  2266          2243  3
;  2267          2244  3 LOCAL
;  2268          2245  3     status;
;  2269          2246  3
;  2270          2247  3 outrab [rab$w_rsz] = .len;
;  2271          2248  3 outrab [rab$l_rbf] = .buf;
;  2272          2249  3
;  2273          2250  4 IF NOT (status = $PUT(RAB = outrab))
;  2274          2251  3 THEN
;  2275          2252  4     BEGIN
;  2276          2253  4
;  2277          2254  4     !+
;  2278          2255  4     ! if the error is due to a record which was too long, shorten the
;  2279          2256  4     ! request and try again
;  2280          2257  4     !-
;  2281          2258  5     IF (
;  2282          2259  6           (.max_rec GTR 80)           ! we aren't pretty short already
;  2283          2260  5         AND
;  2284          2261  6           (
;  2285          2262  7               (.status EQL RMS$_RSZ)  ! error is rec too big (get from tape)
;  2286          2263  6                OR
;  2287          2264  7               (
;  2288          2265  8                   (.status EQL RMS$_SYS)      ! terminal maxbuf error
;  2289          2266  7                    AND
;  2290          2267  8                   (.outrab[rab$l_stv] EQL SS$_EXQUOTA)
;  2291          2268  7               )
;  2292          2269  6           )
;  2293          2270  5         )
;  2294          2271  4     THEN
;  2295          2272  5         BEGIN
;  2296          2273  5         max_rec = (.len * 90) / 100;    ! try with rec 90% as long
;  2297          2274  5         put (.max_rec, .buf);
;  2298          2275  5         RETURN;
;  2299          2276  5         END
;  2300          2277  4     ELSE
;  2301          2278  4         file_error (srh$_writeerr, .status, outfab, .outrab[rab$l_stv]);
;  2302          2279  3     END;
;  2303          2280  3
;  2304        L 2281  3 %IF switch_statistics
;  2305          2282  3 %THEN
;  2306          2283  3     stat_totput = .stat_totput+1; ! Count this line
```

```
;  2307          2284  3 %FI
;  2308          2285  3
;  2309          2286  3 RETURN;
;  2310          2287  2 END;


                                                                    .EXTRN  SYS$PUT

                                                 000C 00000 PUT:    .WORD   Save R2,R3                          ; 2241
                                  53      0000'  CF   9E 00002      MOVAB   MAX_REC, R3                        ;
                                  52      0000'  CF   9E 00007      MOVAB   OUTRAB+12, R2                      ;
                              16  A2        04   AC   B0 0000C      MOVW    LEN, OUTRAB+34                     ; 2247
                              1C  A2        08   AC   D0 00011      MOVL    BUF, OUTRAB+40                     ; 2248
                                            F4   A2   9F 00016      PUSHAB  OUTRAB                             ; 2250
                    00000000G     00              01   FB 00019      CALLS   #1, SYS$PUT                        ;
                                  4D              50   E8 00020      BLBS    STATUS, 3$                         ;
                    00000050      8F              63   D1 00023      CMPL    MAX_REC, #80                       ; 2259
                                                  32   15 0002A      BLEQ    2$                                 ;
                    000186A4      8F              50   D1 0002C      CMPL    STATUS, #100004                    ; 2262
                                                  0E   13 00033      BEQL    1$                                 ;
                    0001C10C      8F              50   D1 00035      CMPL    STATUS, #114956                    ; 2265
                                                  20   12 0003C      BNEQ    2$                                 ;
                                  1C              62   D1 0003E      CMPL    OUTRAB+12, #28                     ; 2267
                                                  1B   12 00041      BNEQ    2$                                 ;
                          51      04   AC 0000005A  8F   C5 00043 1$: MULL3   #90, LEN, R1                     ; 2273
                          63           51 00000064  8F   C7 0004C      DIVL3   #100, R1, MAX_REC                  ;
                                            08   AC   DD 00054      PUSHL   BUF                                ; 2274
                                                 63   DD 00057      PUSHL   MAX_REC                            ;
                              A3  AF                02   FB 00059      CALLS   #2, PUT                            ;
                                                    04 0005D        RET                                        ; 2272
                                                 62   DD 0005E 2$:  PUSHL   OUTRAB+12                         ; 2278
                                        A4       A2   9F 00060      PUSHAB  OUTFAB                             ;
                                                 50   DD 00063      PUSHL   STATUS                             ;
                                    00D710D4     8F   DD 00065      PUSHL   #14094548                          ;
                                FDB7    CF               04   FB 0006B      CALLS   #4, FILE_ERROR             ;
                                            0CFC     C3   D6 00070 3$:  INCL    STAT_TOTPUT                    ; 2283
                                                          04 00074      RET                                    ; 2287

; Routine Size:  117 bytes,    Routine Base:  $CODE$ + 0E7F


;  2311          2288  2
;  2312          2289  2 IF .out_file_open
;  2313          2290  2 THEN
;  2314          2291  3         BEGIN
;  2315          2292  3         LOCAL
;  2316          2293  3             len,                        ! Local length
;  2317          2294  3             ptr,                        ! Pointer to current part of buffer.
;  2318          2295  3             printed;                    ! Flag that first part printed
;  2319          2296  3
;  2320          2297  3         len = .length;                  ! Copy inputs to local
;  2321          2298  3         ptr = .buffer;
;  2322          2299  3         printed = false;
;  2323          2300  3         !
;  2324          2301  3         ! Print the record.  We must allow for a segmented put if the
;  2325          2302  3         ! size of the record is larger than the MRS of the output file.
```

```
;  2326      2303  3          !
;  2327      2304  3          DO
;  2328      2305  4              BEGIN
;  2329      2306  4              IF .printed
;  2330      2307  4              THEN
;  2331      2308  4                  put (11, UPLIT BYTE('(continued)'));
;  2332      2309  4              put (MIN(.len, .max_rec), .ptr);
;  2333      2310  4              len = .len - .max_rec;
;  2334      2311  4              ptr = .ptr + .max_rec;
;  2335      2312  4              printed = true;
;  2336      2313  4              END
;  2337      2314  3          UNTIL .len LEQ 0;
;  2338      2315  2          END;
;  2339      2316  1 END;


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                29 64 65 75 6E 69 74 6E 6F 63 28  002B4 P.AHF:  .ASCII  \(continued)\                          ;


                                                        .PSECT  $CODE$,NOWRT,2

                                          54  DD 00000 SRH$PUT_OUTPUT:
                                                       PUSHL   R4                                             ; 2227
                              3A    0000'   CF  E9 00002         BLBC    OUT_FILE_OPEN, 4$                   ; 2289
                              54            53  D0 00007         MOVL    BUFFER, PTR                         ; 2298
                                            53  D4 0000A         CLRL    PRINTED                             ; 2299
                              0B            53  E9 0000C 1$:     BLBC    PRINTED, 2$                         ; 2306
                                    0000'   CF  9F 0000F         PUSHAB  P.AHF                               ; 2308
                                            0B  DD 00013         PUSHL   #11                                 ;
                          FF71  CF          02  FB 00015         CALLS   #2, PUT                             ;
                                            54  DD 0001A 2$:     PUSHL   PTR                                 ; 2309
                              50    0000'   CF  D0 0001C         MOVL    MAX_REC, R0                         ;
                                            52  DD 00021         PUSHL   LEN                                 ;
                              50            6E  D1 00023         CMPL    (SP), R0                            ;
                                            03  15 00026         BLEQ    3$                                  ;
                              6E            50  D0 00028         MOVL    R0, (SP)                            ;
                          FF5B  CF          02  FB 0002B 3$:     CALLS   #2, PUT                             ;
                              52    0000'   CF  C2 00030         SUBL2   MAX_REC, LEN                        ; 2310
                              54    0000'   CF  C0 00035         ADDL2   MAX_REC, PTR                        ; 2311
                              53            01  D0 0003A         MOVL    #1, PRINTED                         ; 2312
                                            52  D5 0003D         TSTL    LEN                                 ; 2314
                                            CB  14 0003F         BGTR    1$                                  ;
                                            10  BA 00041 4$:     POPR    #^M<R4>                             ; 2316
                                                05 00043         RSB                                         ;

; Routine Size:  68 bytes,    Routine Base:  $CODE$ + 0EF4
```

```
; 2341          2317  1 END
; 2342          2318  0 ELUDOM


                                                        .EXTRN  LIB$SIGNAL, LIB$STOP

;                               PSECT SUMMARY
;
;       Name                    Bytes                   Attributes
;
;  $OWN$_DZRO                     4940  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $OWN$_CRF                      1821  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                          703  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                         3896  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                              Library Statistics
;
;                                       -------- Symbols --------      Pages       Processing
;       File                            Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1      9776       108         1         581        00:01.0



;                                       COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:SEARCH/OBJ=OBJ$:SEARCH MSRC$:SEARCH/UPDATE=(ENH$:SEARCH)

; Size:          3896 code + 7464 data bytes
; Run Time:         01:19.6
; Elapsed Time:     01:29.4
; Lines/CPU Min:     1748
; Lexemes/CPU-Min: 23398
; Memory Used:  451 pages
; Compilation Complete
```

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

LUV1ROM

PQBTDRIVR
LIS

QVSS
LIS

SEARCHMSG
LIS

VMBUVAX1
MAP

SETUSER
LIS

VMBUVAX1
LIS

BOOTDRUV1
LIS

FILERDUV1
LIS

BOOTIOUV1
LIS

CONIO
LIS

NETBOOT
LIS